KAFE: Automated Feature Enhancement for Predictive Modeling using External Knowledge

Sainyam Galhotra[1*] **Udayan Khurana**[2] **Oktie Hassanzadeh**[2]
Kavitha Srinivas[2] **Horst Samulowitz**[2]
[1]University of Massachusetts Amherst, [2]IBM Research
sainyam@cs.umass.edu,{ukhurana, hassanzadeh, kavitha.srinivas, samulowitz}@us.ibm.com

Abstract

The efficacy of a supervised learning model depends on the predictive ability of the underlying features. A crucial means of improving the model quality is to add new features with additional predictive power. This is often performed by domain specialists or data scientists. It is a complex and time-consuming task because the domain expert needs to identify data sources for new features, join them, and finally select the features that are relevant to the prediction. We present a new system called *KAFE (Knowledge Aided Feature Engineering)*, that helps build strong predictive models by automatically performing feature enhancement. It utilizes structured knowledge present on the web, such as knowledge graphs, web tables, etc., and figures out additional information that can improve the accuracy of predictive models. In this paper, we describe the key aspects of the system such as feature inference and selection, along with relevant data indexing for numerical and categorical features.

1 Introduction

In recent years, we have witnessed proliferation of predictive modeling applications in various domains. The cornerstone of building effective models is successful feature engineering. It is a lengthy and time-consuming process that relies on the domain knowledge of the data scientist, and often accounts for up to 80% of the time involved in building predictive models[2]. For instance, when building a predictive model of sales at a store, seasonal holidays such as Christmas are important to encode as features, because they account for surges in sales. Similarly, weather can dampen store sales at a brick and mortar store, and so could represent a key feature for predicting sales.

Often, the domain expertise needed for the addition of such features is not available or is prohibitively expensive to obtain. Moreover, even when it is available, the addition of features is mostly a human-driven process. It is tedious due to the guesswork, coding, and involves a lengthy trial and error process. However, lack of investment in feature engineering impacts the quality of models and consequently, results in a lack of confidence in the use of such models in the real world. Automation of these time-consuming aspects of a machine learning cycle is therefore an area of significant importance and recent interest. Different approaches to perform automated feature engineering have been proposed recently and are being used in the industry[3]. Recent works on feature engineering or enhancement [7, 9, 6, 11] have proposed different methods for *transforming* the feature space using mathematical functions in order to morph the data into a more suitable feature set. Similarly, Deep Learning performs feature engineering implicitly by embedding features based on learned

[*] Work done during first author's internship at IBM Research.

[2] https://tinyurl.com/yyb83ujh

[3] https://www.featuretools.com/, https://www.h2o.ai, https://www.ibm.com/cloud/watson-studio/autoai, https://cloud.google.com/automl-tables/

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

functional mappings. However, most of these techniques are oblivious to the semantic understanding of the features in data. Moreover, they do not consider additional information available from various external sources. Our focus is to fill that gap; specifically, we focus here on the problem of finding **new** data sets and semantically informative features relevant to the predictive model at hand, and adding those that have predictive value automatically into the model. We believe, this is the first work that considers addition of features having semantic explainable relationships with the input data and the techniques presented in this paper provide a valuable addition to the problem of automated feature engineering.

The web contains a plethora of expert knowledge in the form of encyclopedic data, books, blogs, dialogs, and more. A part of this information exists as structured knowledge, in the form of knowledge graphs, ontologies, web tables amongst other sources. Given the recent progress of building knowledge graphs for various domains to capture domain knowledge, devising a system to leverage this high quality structured information is of utmost importance. This paper presents a system called KAFE that aims to utilize this information to improve classification and regression problems.

Given the massive scale of information present on the web coupled with heterogeneity of information available from different sources, automatic identification of relevant knowledge for predictive modeling is not trivial. This work presents novel techniques to identify potential matches of the given data set with structured information on the web and quickly identifies the smallest set of features that help to boost the quality of a classifier. Apart from the novel techniques, this paper also attempts to familiarize the reader with usability the KAFE system. It specifically focuses on providing a handle to the end-user for better understanding of the performance of different components and the ability to plug in different techniques to monitor model performance. One of the major challenges of building our system was to deal with the information available from various data sources like DBpedia, Wikidata and other web tables (more than 20 million) in a unified manner. To this end, we present a *Semantic Feature Index* in KAFE, that provides the dual benefit of achieving fast lookup speed (less than 1 millisecond to identify one entity match) and knowledge inference from millions of data sources are some of the notable strengths of KAFE.

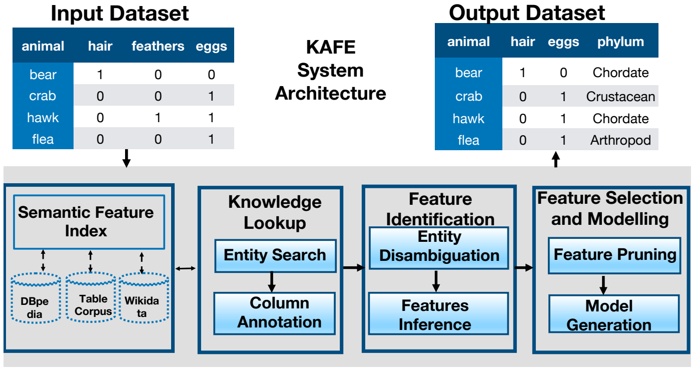
2 Related Work

Feature engineering has traditionally been a manual process. Recently, there has been some work to automate it. For instance, Cognito [9, 10] performs trial and error through reinforcement learning to find suitable "transformation functions" that generate new features from base features. Katz et al. [7] rely on a rather exhaustive algorithm that greedily evaluates many choices for feature transformations. LFE [11] learns patterns of associations between features, transforms and target from historical data to predict transforms for new data. All of these methods do not add new information that is external to the given data. A detailed account of such automated feature engineering techniques can be found here [8].

Friedman et al. [2] recursively generate new features using a knowledge base for text classification. FeGeLOD [14] extracts features from Linked Open Data via queries. These approaches, however, fail to thoroughly explore the available information from external web sources, and apply it to general machine learning problems. In contrast, our system automatically infers and selects novel and useful features from external sources such as web tables to boost the predictive performance. The problem of assigning semantic types to columns of tabular data is closely related to the objective of feature enhancement. For categorical column values, most approaches annotate them by exact matching to KB instances, classes, and properties. Systems such as ColNet [1] and Sherlock [4] exploit and extract contextual semantics from the tabular data using neural networks. Compared to textual data, numerical data is much harder to annotate. Most previous works develop a background KB from sources like Wikidata and then apply approaches such as transformation [13] and k-nearest neighbors search [12] to collect candidate semantic labels. These approaches are complimentary to the ideas described in this paper, and can enhance our pipeline in the data integration step. An earlier version of KAFE that worked only on categorical attributes can be seen in [3].

3 System Description

Figure 1 illustrates an overview of our system. The main highlight of KAFE is that it provides a simplified access to the structured knowledge present in various knowledge graphs, as well as

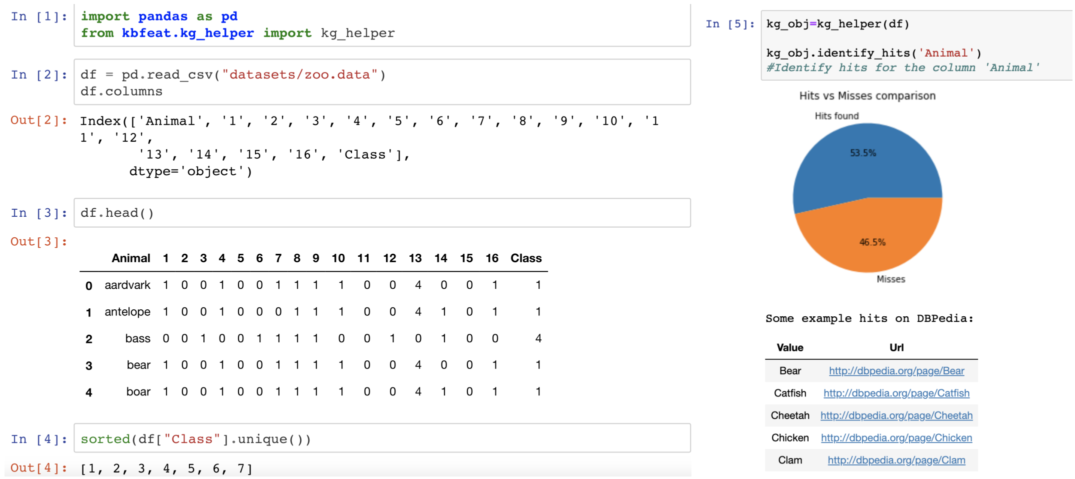


animal	hair	feathers	eggs
bear	1	0	0
crab	0	0	1
hawk	0	1	1
flea	0	0	1

animal	hair	eggs	phylum
bear	1	0	Chordate
crab	0	1	Crustacean
hawk	0	1	Chordate
flea	0	1	Arthropod

Figure 1: KAFE System Architecture

web sources along with an ability for the end-user to plug in their proprietary data. KAFE consists of state-of-the-art techniques to process the input data set, generate insightful visualizations and automatically identify the minimal set of features that provide maximum gain to model performance. KAFE provides transparency to inspect the internals of various algorithms. It consists of the following key components:



```
In [1]: import pandas as pd
        from kbfeat.kg_helper import kg_helper

In [2]: df = pd.read_csv("datasets/zoo.data")
        df.columns

Out[2]: Index(['Animal', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '1
        1', '12',
               '13', '14', '15', '16', 'Class'],
              dtype='object')

In [3]: df.head()

Out[3]:
```

	Animal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Class
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
1	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	1
2	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	4
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	1
4	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	1

```
In [4]: sorted(df["Class"].unique())

Out[4]: [1, 2, 3, 4, 5, 6, 7]
```

```
In [5]: kg_obj=kg_helper(df)

        kg_obj.identify_hits('Animal')
        #Identify hits for the column 'Animal'
```

```
Some example hits on DBPedia:
```

Value	Url
Bear	http://dbpedia.org/page/Bear
Catfish	http://dbpedia.org/page/Catfish
Cheetah	http://dbpedia.org/page/Cheetah
Chicken	http://dbpedia.org/page/Chicken
Clam	http://dbpedia.org/page/Clam

(a) Data set statistics

(b) Data-DBpedia Mapping

Figure 2: The Python front-end helps to easily load a data set, explore the different features and pre-process. The library is easy to invoke and it quickly shows statistics of the number of matches found between the cells of a column with the external knowledge graph. This snapshot shows the identified hits for Zoo data set (`https://archive.ics.uci.edu/ml/datasets/Zoo`).

- **Knowledge lookup**: This component iterates over different cells of the input and queries the index built on top of DBpedia, Wikidata and a collection of around 25 million web tables. The lookup component tries to identify an exact match initially and if no matches are found, it queries the index for fuzzy matches. KAFE leverages these hits to identify the semantic type of the column by voting, which is helpful for the broader understanding of the column and is useful for cell-level disambiguation in the next phase. KAFE employs sampling strategies to optimize the number of cells queries to the index and achieve better efficiency.

- **Entity Disambiguation**: The knowledge lookup phase generates different candidate hits for each cell in the table. In some cases, we find multiple hits. For example, the animal ‘Duck’ maps to three different entities (i) An animal (`http://dbpedia.org/page/Duck`), (ii) A City in North Carolina (`https://www.wikidata.org/wiki/Q2724064`) and (iii) A Cartoon character (`http://dbpedia.org/page/Donald_Duck`). KAFE disambiguates such scenarios by leveraging the semantic type of the column to construct a precise mapping between the various values in our data set to the closest knowledge graph entities and relationships. After identifying this mapping, KAFE explores the attributes of the mapped entities in knowledge graphs (or tables) to identify different candidate features. We observed that a number of DBpedia properties have multiple attribute values (For example, the animal ‘duck’ has more than four values for property ‘type’: Animal, Bird, Eukaryote, Species, etc. `http://dbpedia.org/page/Duck`).

• **Feature Selection and Modeling**: Once a list of candidate features is identified, KAFE performs eliminates features with few distinct values, low information gain and low correlation with the target class. The subset of features are then used to train a model. The features used to train are further pruned based on their importance values. KAFE provides the flexibility to employ other techniques to identify a small set of useful features and train a predictive model.

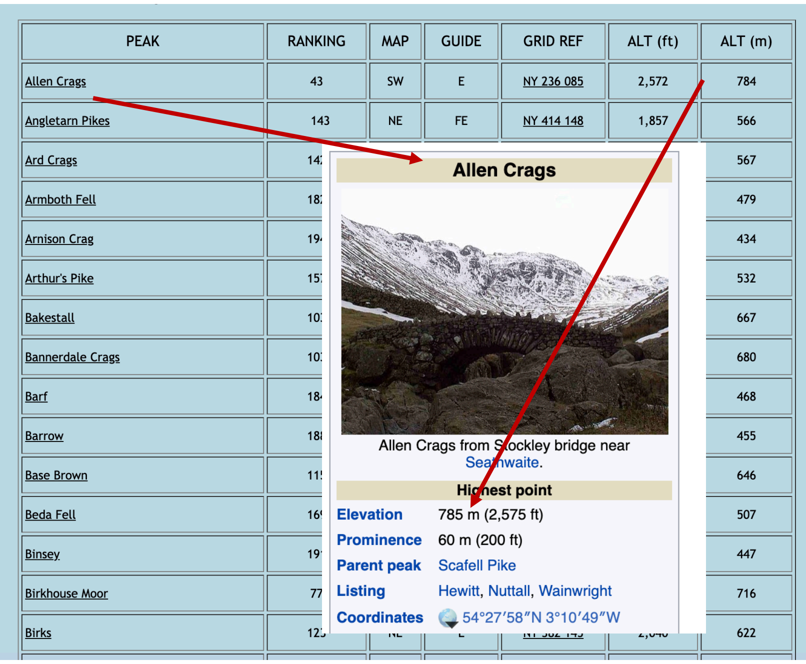
• **Indexing**: Given that the scale of the data available from various knowledge sources is in the millions, providing a quick access to this information was one of the key challenges for our problem. We constructed an index on top of DBpedia, Wikidata data set and a dump of structured data (present in form of tables collected from various sources). In total we consider a collection of around 15 million entities. This index helps in quick lookup of an entity and examine its attributes. Given the heterogeneity of information present in various tables, our index constructs a novel homogeneous representation based on the semantic type of the information present in those data sources. Our index leverages this homogeneous representation to shortlist columns that can be joined, to speed-up the knowledge lookup and feature exploration phase.

There are a number of important challenges that need to be addressed to devise an automatic holistic feature generation pipeline that leverages external knowledge. In this work, we focus on the semantic mapping of columns that contain numerical data and present various techniques devised to handle such scenarios.

4 Joins Based on Numerical Attributes

Knowledge lookup on numerical attributes is a difficult problem because numeric values can match across semantically disparate columns, whereas text tends to be less ambiguous. As an example, *2010* can match to the year that Rafael Nadal won his first US Open, or the monthly salary of some employee. Identifying the correct mapping for numerical attributes therefore requires a different set of techniques than mapping for text. To solve this problem, we devise a suite of techniques which are together helpful to match the different numerical attributes.

- Numerical distribution testing: This approach tries to construct a representation of the different numerical attributes and performs distribution testing to identify the closest distribution. The numerical values for the column are represented as a fixed length vector, where the features are determined by statistical properties such as the mean, variance, minimum, maximum, range and histogram of quantiles. Vectors of column values and knowledge graph values are then compared for L1 and L2 distances. While this approach of breaking down numerical attributes into a set of features is not new (e.g., see [5]), KAFE can combine this the notion of using pivots in the knowledge graph to shortlist the possible column matches for a numerical column. We describe this notion next.
- Pivot based annotation: We stage the matching process such that textual columns which are less ambiguous are matched first, and these textual columns are then used as anchors to severely circumscribe the set of numerical attributes in the knowledge graph that may be potential matches for the numerical columns. Figure 3 shows this idea using an example. At this point, the type of *Peak* has been matched to the knowledge graph type of *Mountain* and the entity has been disambiguated to be *Allen Crags*, a peak in the knowledge graph. The numerical value of *alt* is numerically close to the value of *elevation*, and hence *elevation* is a good candidate match for *alt*.
- Column headers: When possible, we also determine whether column header strings map directly to properties in the knowledge base, either directly, or through synonyms for properties using other knowledge bases such as WordNet. For instance, consider the same dataset in Figure 3, and the column heading *ALT*. Using WordNet, as shown in Figure 4, this column heading can be mapped to the semantic property of *Elevation*. Note that although we use exact matches in this example, this is only for illustration. In reality we use techniques like using word embeddings to improve recall.



PEAK	RANKING	MAP	GUIDE	GRID REF	ALT (ft)	ALT (m)
Allen Crags	43	SW	E	NY 236 085	2,572	784
Angletarn Pikes	143	NE	FE	NY 414 148	1,857	566
Ard Crags	14					567
Armboth Fell	18					479
Arnison Crag	19					434
Arthur's Pike	15					532
Bakestall	10					667
Bannerdale Crags	10					680
Barf	18					468
Barrow	18					455
Base Brown	11					646
Beda Fell	16					507
Binsey	19					447
Birkhouse Moor	77					716
Birks	12					622

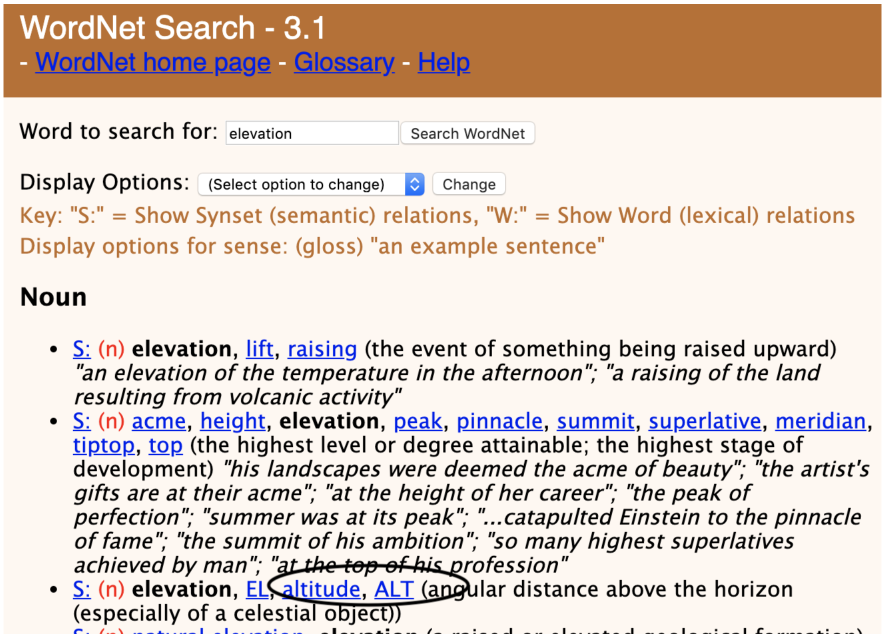
Allen Crags

Allen Crags from Stockley bridge near Seathwaite.

Highest point

Elevation	785 m (2,575 ft)
Prominence	60 m (200 ft)
Parent peak	Scafell Pike
Listing	Hewitt, Nuttall, Wainwright
Coordinates	54°27′58″N 3°10′49″W

Figure 3: Pivot based annotation for numerical columns based on entity matching.



WordNet Search - 3.1

- WordNet home page - Glossary - Help

Word to search for: elevation

Display Options: (Select option to change)

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

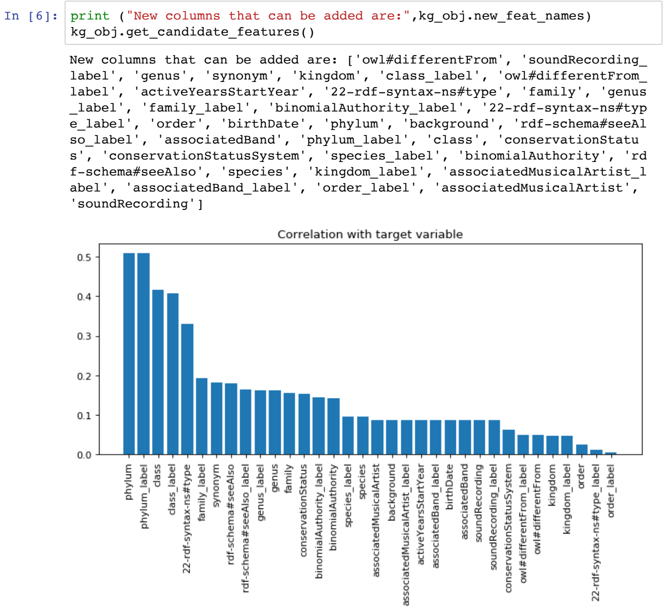
- S: (n) **elevation**, lift, raising (the event of something being raised upward) *"an elevation of the temperature in the afternoon"; "a raising of the land resulting from volcanic activity"*
- S: (n) acme, height, **elevation**, peak, pinnacle, summit, superlative, meridian, tiptop, top (the highest level or degree attainable; the highest stage of development) *"his landscapes were deemed the acme of beauty"; "the artist's gifts are at their acme"; "at the height of her career"; "the peak of perfection"; "summer was at its peak"; "...catapulted Einstein to the pinnacle of fame"; "the summit of his ambition"; "so many highest superlatives achieved by man"; "at the top of his profession"*
- S: (n) **elevation**, EL, altitude, ALT (angular distance above the horizon (especially of a celestial object))

Figure 4: Wordnet to map column headers to knowledge base properties

5 Using KAFE for Predictive Modeling

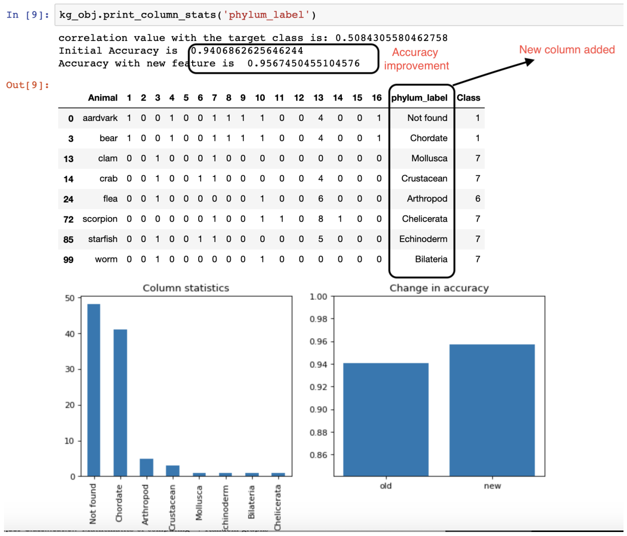
In this section, we familiarize the reader with the usability of KAFE. It can be used through an interactive Jupyter notebook[4]. We will highlight functions to perform the following: (i) analyze the input data set, (ii) visualize and modify the mapping of entities to external knowledge, feature generation and selection, (iii) identify the smallest set of features that achieve the best performance. We will provide five pre-loaded classification and regression data sets for use. The user will go through the following components, with an option to customize or alter the workflow as desired:

[4] Video of our system: `https://youtu.be/7HbHeNYRh_c`



```
In [6]: print ("New columns that can be added are:",kg_obj.new_feat_names)
        kg_obj.get_candidate_features()

        New columns that can be added are: ['owl#differentFrom', 'soundRecording_
        label', 'genus', 'synonym', 'kingdom', 'class_label', 'owl#differentFrom_
        label', 'activeYearsStartYear', '22-rdf-syntax-ns#type', 'family', 'genus
        _label', 'family_label', 'binomialAuthority_label', '22-rdf-syntax-ns#typ
        e_label', 'order', 'birthDate', 'phylum', 'background', 'rdf-schema#seeAl
        so_label', 'associatedBand', 'phylum_label', 'class', 'conservationStatu
        s', 'conservationStatusSystem', 'species_label', 'binomialAuthority', 'rd
        f-schema#seeAlso', 'species', 'kingdom_label', 'associatedMusicalArtist_l
        abel', 'associatedBand_label', 'order_label', 'associatedMusicalArtist',
        'soundRecording']
```

Figure 5: The different features that can be added to the Zoo data set and their correlation with the target variable.



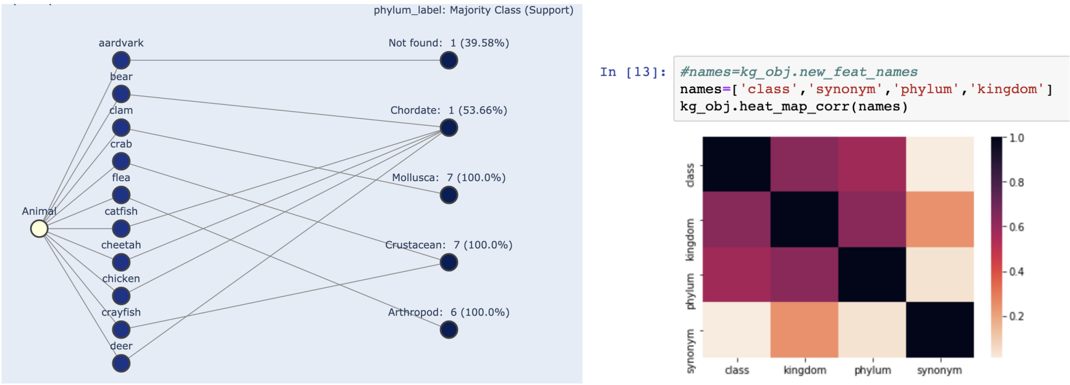
```
In [9]: kg_obj.print_column_stats('phylum_label')

        correlation value with the target class is: 0.5084305580462758
        Initial Accuracy is  0.9406862625646244
        Accuracy with new feature is  0.9567450455104576
Out[9]:
```

	Animal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	phylum_label	Class
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	Not found	1
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	Chordate	1
13	clam	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	Mollusca	7
14	crab	0	0	1	0	0	1	1	0	0	0	0	0	4	0	0	0	Crustacean	7
24	flea	0	0	1	0	0	0	0	0	0	1	0	0	6	0	0	0	Arthropod	6
72	scorpion	0	0	0	0	0	0	1	0	0	1	1	0	8	1	0	0	Chelicerata	7
85	starfish	0	0	1	0	0	1	1	0	0	0	0	0	5	0	0	0	Echinoderm	7
99	worm	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	Bilateria	7

Figure 6: A summary of the 'phylum_label' feature identified, effect on classification accuracy and the correlation with the target class.

1. **Data analysis**: We provide the user, a complete handle on the data set by providing functions and visualization tools to analyze the different features. Our Python notebook allows the user to invoke any of the built-in functions in combination with our tools. The analysis of correlation and feature importance between the different features helps identify ineffective features and streamline the search for relevant domain knowledge that can help boost the classification performance. Figure 2 displays KAFE's UI to help facilitate data loading, external knowledge mapping and analysis.

2. **Knowledge Index lookup**: KAFE provides a one-stop solution to all the data sources including DBpedia, WikiData and a collection of more than 20 million web tables. It provides different functions to query a particular entity and explore the different hits identified. Figure 5 shows the utility of the different features to predict the target class. Figure 6 shows the deeper analysis of the particular feature, the column statistics, and classification improvements. Figure 7(a) shows a small subset of the knowledge graph, zoomed into the property of interest along with statistics about the target variable that is best captured by this property.



```
In [13]: #names=kg_obj.new_feat_names
         names=['class','synonym','phylum','kingdom']
         kg_obj.heat_map_corr(names)
```

(a) Knowledge Graph snapshot

(b) Correlation heat map

Figure 7: The knowledge graph zoomed-in for ‘Phylum’ property of the identified hits along with a summary of the target class being captured by each value and Correlation heat map between the new features identified. Highly correlated features are expected to contribute towards similar examples.

3. **Inference**: This module of KAFE demonstrates the inferred feature and its statistics e.g., correlation with the target variable, number of hits identified and change in accuracy if this feature is considered in the training phase. The different statistics and interactive visualizations generated by KAFE help the end user to quickly understand new candidate features, their summary and usefulness. KAFE exposes functionality for the user to compare the benefit of one feature on top of another by generating a correlation heat map along with change in accuracy. Figure 7(b) shows the correlation values between every pair of new features identified.

4. **Feature Selection and Model Building**: This visualization component displays the accuracy improvement on considering new features identified from external knowledge sources along with feature importance. Due to space constraints, we do not show a screen shot of this aspect of our demonstration.

6 Experiments

We evaluate the end-to-end solution of feature enhancement over real world data sets used for prediction tasks and evaluate the efficacy of semantically matching numerical columns.

We test KAFE on Zoo Animal Classification data set [5] to test the F-score of a classifier trained with and without feature enhancement. We observe that the F-score improved from 0.94 to 0.974 with the help of additional features from DBPedia and Wikipedia. Due to the efficiency of our semantic index, this feature enhancement pipeline was able to identify all the useful features for Zoo data set in less than a minute.

To test the different techniques for labelling numerical attributes, we consider a collection of six tables from T2Dv2 corpus[6] and ISWC 2019 challenge [7]. The different numerical columns in these tables referred to Altitude, Country’s population and Area, Floor count and height of tallest buildings of the world, dimensions of famous lakes around the world, number of passengers flying from different airports, length of a movie and its year of release and the Zoo data set.

We observe that the simplistic approach of calculating L1 distance shows very poor performance because the considered statistics over the distributions match with many distributions. The distribution of country’s population was confused with ‘World Heritage Site ID’ and some other id’s present over the knowledge graph. The technique of leveraging column headers works well for columns with well-defined headers. It fails to identify the column containing volume of water in famous lakes because of noise in column header. Similarly, it did not work for column containing the length of a movie because the column header consisted of length and DBPedia had a label of runtime. The pivot based algorithm worked well for the columns which had noisy or no headers but did not work

[5] https://www.kaggle.com/yunusulucay/zoo-animal-classification-using-ml/output

[6] http://webdatacommons.org/webtables/goldstandardV2.html

[7] http://www.cs.ox.ac.uk/isg/challenges/sem-tab/

for number of passengers flying from different airports. The main reason was the large difference in values of the knowledge graph and that of the table being considered. It is primarily because of temporal aspect of the data. Additionally, Wikidata had an attribute named Patronage with multiple values (one for each year). Current, KAFE does not handle attributes with multiple values. These techniques did not apply for Zoo data set because of binary encoding of different values in the column. Devising techniques to handle such encoding of columns is an interesting extension of this work.

7 Conclusion

In this paper, we presented the techniques and usability details of a novel system called KAFE (Knowledge Aided Feature Engineering). It helps data scientists perform better predictive modeling by utilizing massive amounts of structured knowledge on the web, such as knowledge graphs and web tables. It helps enhance the feature space of a given supervised learning problem by automatically identifying feature types, matching and joining them with external data through a novel index built on web data called the Semantic Feature Index. We hope this work in progress demonstrates one crucial step in the direction of utilizing the massive potential of the semantic web towards effective application of machine learning.

References

[1] J Chen, E Jiménez-Ruiz, I Horrocks, and C Sutton. Colnet: Embedding the semantics of web tables for column type prediction. In *AAAI*, 2019.

[2] L Friedman and S Markovitch. Recursive feature generation for knowledge-based learning. *arXiv preprint arXiv:1802.00050*, 2018.

[3] Sainyam Galhotra, Udayan Khurana, Oktie Hassanzadeh, Kavitha Srinivas, Horst Samulowitz, and Miao Qi. Automated feature enhancement for predictive modeling using external knowledge. *IEEE ICDM*, 2019.

[4] M Hulsebos, K Hu, M Bakker, E Zgraggen, A Satyanarayan, T Kraska, Ç Demiralp, and C Hidalgo. Sherlock: A deep learning approach to semantic data type detection. *KDD*, 2019.

[5] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. Sherlock: A deep learning approach to semantic data type detection. KDD '19, 2019.

[6] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *IEEE DSAA*, 2015.

[7] G Katz, E Shin, and D Song. Explorekit: Automatic feature generation and selection. In *IEEE ICDM*, 2016.

[8] U Khurana. Transformation-based feature engineering in supervised learning: Strategies toward automation. In *Feature Engineering for Machine Learning and Data Analytics*. 2018.

[9] U Khurana, H Samulowitz, and D Turaga. Feature engineering for predictive modeling using reinforcement learning. *AAAI*, 2018.

[10] U Khurana, D Turaga, H Samulowitz, and S Parthasarathy. Cognito: Automated feature engineering for supervised learning. In *IEEE ICDM*, pages 1304–1307, 2016.

[11] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana, Elias B. Khalil, and Deepak Turaga. Learning feature engineering for classification. In *IJCAI*, 2017.

[12] S Neumaier, J Umbrich, JX Parreira, and A Polleres. Multi-level semantic labelling of numerical values. In *ISWC*, 2016.

[13] P Nguyen and H Takeda. Semantic labeling for quantitative data using wikidata. 2018.

[14] H Paulheim and J Fümkranz. Unsupervised generation of data mining features from linked open data. *WIMS*, 2012.