Feature Engineering (FE) Tools and Techniques for Better Classification Performance

Article · May 2019

DOI: 10.21172/ijiet.82.024

Tara Rawat
JSS Academy of Technical Education
1 PUBLICATION **2** CITATIONS
SEE PROFILE

Vineeta Khemchandani
JSS Academy of Technical Education
15 PUBLICATIONS **56** CITATIONS
SEE PROFILE

International Journal of Innovations in Engineering and Technology (IJIET)
http://dx.doi.org/10.21172/ijiet.82.024

Feature Engineering (FE) Tools and Techniques for Better Classification Performance

Tara Rawat
Department of Computer Applications
JSS Academy of Technical Education, Noida, India

Dr. Vineeta Khemchandani
Department of Computer Applications
JSS Academy of Technical Education, Noida, India

Abstract - Feature engineering has been the focus of interest for some time and it is still limited or under studied. Therefore, more determined attempts are required to help forward feature engineering process in the context of learning algorithms to predict better results and behaviours. With the huge amount of data available and the consequent requirements for Artificial Intelligence and good machine learning techniques, new problems arise and novel approaches to feature engineering techniques are in demand. This paper presents a comprehensive survey of methodologies, tools and techniques used for feature engineering with the purpose of improving model(classifier) accuracy on unseen data and also presents applications of feature engineering in text classification, clinical text classification, link prediction on social networks, knowledge base construction, fraud detection and other domains, used to achieve high performance of predictive learning algorithms in terms of model accuracy.

Keywords: Feature Engineering, Machine learning, Classifier, Predictive learning

I. INTRODUCTION

In classifying objects, the large number of features(attributes) indicates the complexity of a problem. To solve complex classification problem, good features plays a crucial role, which have a higher predictive power. Though, several dimensional reduction techniques like feature selection exist to help many aspects of learning classification problems, but so far there is little support for a crucial step in the process of engineering features. Engineering good features set is prerequisite to achieve high accuracy in classifying objects and prediction.

As classification systems become an integral part of many domains to take effective decisions or solving prediction problems with least error rate and highest accuracy. It is difficult, however, to compare the accuracy of the techniques and determine the best one because their performance is data(feature) dependent. It is therefore necessary to move towards adaptive classification systems that selectively employ appropriate classification method(s) by first engineering the available raw data as features that are basic building blocks or an observation for describing data to an algorithm(classifier) that implements classification or prediction(e.g. pattern recognition - handwriting, speech, face recognition, Internet applications- text categorization and medical applications-diagnosis, drug discovery, prognosis etc.) . Features must capture salient aspects of the variables to help the target function learn the target result (e.g. health status of new patients: healthy or suffering from cancer). But features may deviate the results if they insufficiently describe the underlying problem or are not impactful for the problem. Although there are many pre –processing steps for the inputs to be fed into an algorithm(classifier) for the learning process (e.g. data selection, dimension reduction, data tuning etc.), both data scientist[1] and analytics[2] find feature engineering as difficult to the learning process. Features variety may be 1) strongly relavant: always preferred 2) weakly relevant: these may be combined to improve relevance for the problem 3) irrelevant: must be discarded and 4) redundant: removed or ignored[3]. Human involevemt is required in feature engineering to improvre the performance[4]. Feature engineering is a process which helps to generate features that are general and flexible, so they can be reused in other models for other purposes. Trained systems are very challenging to build and picking up the right features for training the system is the most critical and time consuming part in developing a successful classification or prediction model.

This paper discusses about the role of features in classification problems. The remainder of the paper is organized as follows, in section II, related work and feature engineering for classification problems and its process have been discussed. The generic feature engineering methods have been listed in section III. The existing feature engineering tools and different applications of feature engineering for classification task has been briefed in section IV and section V respectively. Section VI concludes the research on feature engineering.

II. BACKGROUND AND RELATED WORK

In this section, we consider the challenge of creating new features and review research on feature engineering in classification problems.

A. Features in learning Models

Good feature is a key problem in learning classification/predictive models as it can largely determine the performance of the underlying problem. A good feature should be 1) informative 2) never change for the set of transformations and 3) fast to compute. Data analytics often begin by exploring the existing features from the underlying problem, using their own knowledge domain to think of appropriate features or trying to algorithmically generate features for the given data set [5].

There have been many algorithms for speech recognition, real time sensors etc. where automatically new features can be build out of the raw data and many techniques available for selecting features to improve efficiency and generalization but there is no optimized solution which can be used across data sets.

B. Feature Engineering(FE) for classification Problems

FE is foundation to the learning algorithms. It is the process of using domain knowledge of the data to create features that make learning algorithm works[6]. In order to work predictive/classified learning algorithms for underlying problems, inputs must be transformed/turned into such format so that algorithm understands it and gives the accurate class to which an entity belongs and future prediction. It is typically the most time consuming part in building classifiers. For an instance, predictive learning on images, pixels values are the features, but a higher level of feature engineering can be done to represent it in an more informative manner to feed into the classifier(learning algorithm) like the average pixel intensity, detect the edges and no. of edges exist in the top quadrant of the image etc. These numbers along with the pixel value, get fed into classification algorithm, the task of classifying the image is simplified and the better results will be produced [7].

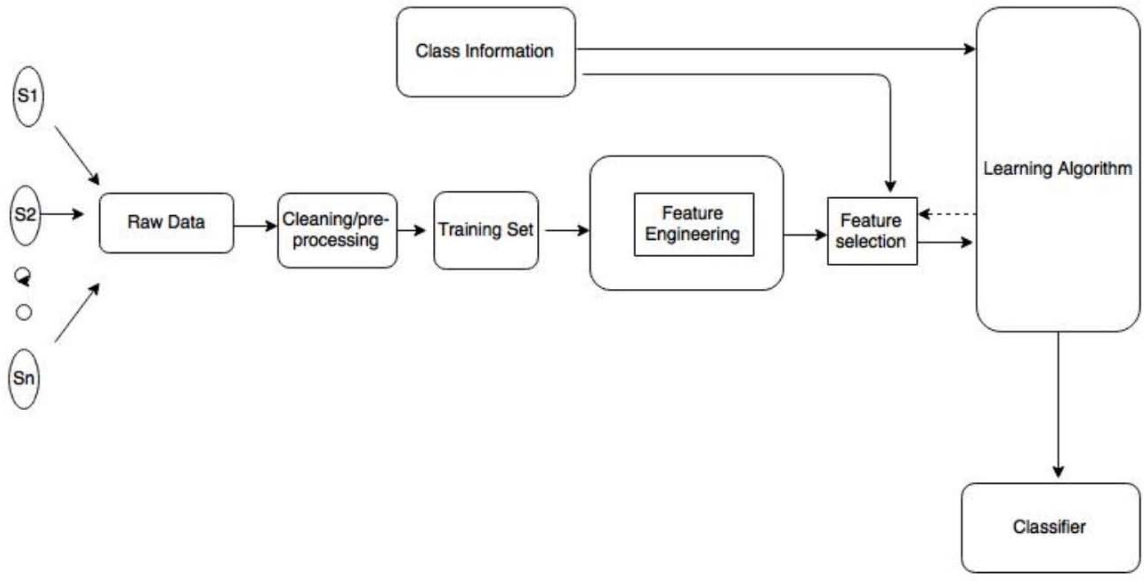

Fig1. A General Framework of Feature Engineering for Classification

Feature Engineering is performed after data scrubbing and preparation, before training the model. Primarily objective is to provide better representation of the data to the predictive learning algorithm.

Step 1: Collection of Raw data : Collection of data from different sources which may be structured, unstructured or textual data.
Step 2: Data Preprocessing: Format the raw data by aligning, unions, grouping, intersections. Removing noisy, dirty data(missing, duplicates, ill-formed, wrong values etc.) by pairwise or listwise deletion, by computing impution(mean substitution, Regression), by stochastic simulation, principle of least harm, sampling error, population parameters, propogation, statistical power etc. as approaches vary quite a lot. Some methods are elabored in the next sext section.
Step 3: Feature Engineering (Feature Transformation and Creation of additional features): Clean data is still raw where lot of data is useless, so data is filtered, sliced and transformed to create features for modeling. Feature engineering is hard where analysis and domain knowledge is required. Some common feature engineering methods might be PCA, Kernal PCA, Partial least square, Discrtization, Information Entropy theory, ICA, MDA, Latent Factors, Statistical Moments, Mutual information theory, Generalized least square, De-Noising, patch extraction, auto encoding, edge detection, wieghting, smooting etc.
Step 4: Selection of Features: Append additional features with training data sets and perform feature selection and extraction and then process the selected features to the learning algorithm. The feature engineering and selection phase may be independent of the learning algorithm.
Step 5: Modeling and performance measure: Create models and it may iteratively utilize the performance of the learning algorithms to evaluate the quality of the selected features, like wrapper models, cross validation etc.
Step 6: Classifier: Finally selected features, a classifier is induced for the classification or prediction phase.

The Data Scientist can use several techniques such as correlations, decision trees, factor analysis etc. as mathematical function to support in the feature engineering process. Besides these traditional techniques that might be inefficient within feature engineering, we need to devise some advancements such as in-memory processing to speed up the complex mathematical operations etc.

III. FEATURE ENGINEERING METHODS

Prior to feature engineering methods, raw data need to be preprocessed as preprocessing also affects performance of classifiers. Some of the data preprocessing methods are as follows:

A. Variable Identification: Identify input and output(target) features(variable), their data type and category. For an instance, inputs are like height, weight, gender output may be PlayCricket. Data type like character, Numeric, Variable category like Categorical, Continuous.

B. Univariate Analysis: Exploring features one by one with their type. If the type is continuous, can be measured using various statistical metrics visualization methods and for categorical, frequency table may be used to understand distribution of each category.

C. Bi-variate Analysis: Finding out relationship between two feature values of different category types.

a) Continuous & Continuous: scatter plots can be used to show the relationship. The strong connection between two feature values can be determined using Correlation, varies between -1 and +1.
Correlation can be derived using equation 1.1
Correlation = Covariance(X,Y) / SQRT(Var(X)* Var(Y)) 1.1 b)
Categorical & Categorical: To find the relationship between two categorical variables, we can use following methods:

- Two-way table: The rows represents the category of one feature value and the columns represent the categories of the other feature value.
- Stacked Column Chart: A visual form of Two-way table.

Chi-Square Test: The chi-square test statistic for a test of independence of two categorical variables is given in equation 1.2:

$$X^2 = \sum (O - E)^2 / E \qquad 1.2$$

where O represents the observed frequency. E is the expected frequency under the null hypothesis and computed by equation 1.3:

$$E = \frac{rowtotal * \text{columntotal}}{samplesize} \qquad 1.3$$

D. Categorical & Continuous: We can use draw box plots for each level of categorical variables.The statistical significance will not be seen for small number of levels. For the statistical significance we can perform Z-test, T-test or ANOVA.

Z-Test/ T-Test:- Either test assess whether mean of two groups are statistically different from each other or not.

$$z = \frac{|\overline{x_1} - \overline{x_2}|}{\sqrt{\frac{s_1{}^2}{n_1} + \frac{s_2{}^2}{n_2}}} \qquad 1.4$$

As in equation 1.4, if the probability of z is small then the difference of two averages is more significant. The T-test is very similar to Z-test but it is used when number of observation for both categories is less than 30.

$$\boxed{\begin{array}{l} t = \frac{\overline{X_1} - \overline{X}_2 |}{\sqrt{S^2(\frac{1}{N_1} + \frac{1}{N_2}}} \\ S^2 = \frac{(N_1 - 1)S_1{}^2 + (N_2 - 1)S_2{}^2}{N_1 + N_2 - 2} \end{array}} \qquad 1.5$$

Where:

- $\overline{X_1}$, $\overline{X}_2$: Averages
- $S_1{}^2$, $S_2{}^2$: Variances
- N_1, N_2 : Counts
- T: has t distribution with $N_1 + N_2 - 2$ degree of freedom

ANOVA:- It assesses whether the average of more than two groups is statistically different.

E. Missing values treatment: There are various methods to treat missing values like deletion, Mean/mode/Median imputation, KNN Imputation etc.

F. Outlier treatment: Its an observation which is detached from an overall pattern which may increase the error variance and decrease the power of statistical tests. Outliers can be detected via various visualization techniques like Box-plot, histogram, Scatter plot and Outliers can be removed via deletion or feature value transformation, Imputation and treat them explicitly.

Some of the Generic Feature Engineering Methods are discussed in this sub section.The Feature Engineering is divided in 2 steps: Feature transformation and Feature creation

A. Feature Transformation: With the target of transforming effective features via FE process in the training data, we bring up effective representation of data. Feature Engineering requires transformation of variables into formatted features that our models can use. Some generic feature transformation methods are discussed here before training the classifiers for the classification task.

1. Basic variable transformation like calculating a new feature value from a single existing feature value. Some instances are singular feature value transformations are as mentioned in equation 1.6:

$$\boxed{\begin{array}{l} y = x^2 \\ y = \sqrt{x} \\ b = \ln a, or \\ yquad = 4y^2 + 3y + 6 \end{array}} \qquad 1.6$$

Some transformations to calculate a new feature value using several existing values mentioned in equations 1.7.

$$\boxed{\begin{array}{l} avg = \dfrac{p+q+r}{3} \\ invsum = \dfrac{1}{x} + \dfrac{1}{y} + \dfrac{1}{z} \\ \text{var}\max = \max of\,(x_1, x_2, x_3) \end{array}} \qquad 1.7$$

2. Feature Scaling might be used, so that all features get normalized. Some methods might be
Rescaling- To scale features range in [0,1] or[-1,1]. The general formula is in equation 1.8:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad 1.8$$

Where x is the normal value and x' is the normalized value.
3. Standardization- A standardized value z is defined as equation 1.9

$$\text{Z} = \log(y) \qquad 1.9$$

where x is the original variable, $\bar{x}$ is the mean of the original variable, and s_x is the standard deviation of the original variable.

Or

To scale the components of a feature vector such that the complete vector has length one. Like provided in equation 1.10

$$x' = \frac{x}{\|x\|} \qquad 1.10$$

Here, Scalar metric is used as a distance measure.
4. To normalize feature values, Box-Cox transformation may be used. The mathematical equation for the transform is given in equation 1.11:

$$\boxed{\begin{array}{l} Y = \dfrac{(X+c)^\lambda - 1}{\lambda}, \lambda \neq 0 \\ Y = \log(X+c), \lambda = 0 \end{array}} \qquad 1.11$$

Here, c is an arbitrary constant chosen so that all scores (i.e., X +c) are greater than 0. The value of λ used in this equation is the one that transforms the data closest to normality and must be found using computer algorithms. It is particularly useful family of transformations and defines as in equation 1.12:

$$T(Y) = (y^\lambda - 1) / \lambda \qquad 1.12$$

where Y is the response variable and λ is the transformation parameter. For λ = 0, the natural log of the data is taken as $\log(y)$.
It will be useful to remove spurious interactions and to identify some significant factors.

5. Min-Max normalization: it performs a linear transformation on the raw data set. Lets say, $\min_x$ and $\min_x$ are the minimum and maximum values of an attribute. It maps a value v_i of X to v_i' in the range [new_ $\min_x$, new_ $\max_x$] by computing as in equation 1.13:

$$v_i' = \frac{v_i}{10^j} \qquad 1.13$$

Min-Max normalization saves the relationships anomg the original feature valuesand throws "out of bounds " error if the range value is outside the original data range of X.

Normalization by decimal scaling:- Normalizes by moving the decimal points of values of attribute X. The number of decimal points moved depends on the maximum absolute value of X. A value, v_i, of X is normalized to v_i' by computing as in equation 1.14:

$$v_i' = \frac{v_i}{10^j} \qquad 1.14$$

Where j is the smallest integer such that max(| v_i' |) <1.

6. The log-odds(logit) feature transformation- In logic regression, the dependent variable is a logit as given in equation 1.15:

$$\log(odds) = \log it(P) = \ln(\frac{P}{1-P}) \qquad 1.15$$

Logit is a log of odds and odds are function of P, the probability of a 1. In logic regression it is as given in equation 1.16:

$$\log it(P) = a + bX \qquad 1.16$$

It is assumed to be linearly related to X, if it is so then the relation between X and P is nonlinear and has the shape of S-shape curve and function form will be as in equation 1.17:

$$\boxed{\begin{array}{l} \ln(\frac{P}{1-P}) = a + bX \\ \frac{P}{1-P} = e^{a+bX} \\ P = \frac{e^{a+bX}}{1+e^{a+bX}} \end{array}} \qquad 1.17$$

7. For high cardinality, construct a feature which captures the frequency of the occurrence of each level of the categorical values.

When the target attribute Y is binary, $Y \varepsilon \{0,1\}$, the transformation maps individual values X_i of a high-cardinality categorical attribute X to a scalar, S_i, representing an estimate of the probability of $Y = I$ given that $X = X_i$ where X_i stands to as in equation 1.18:

$$X_i \rightarrow S_i \cong P(Y \mid X = X_i) \qquad 1.18$$

Here, S_i represents a probability, the transformed attributeS is automatically normalized between 0 and 1, which might be useful feature for netural network models.

8. Discretization: The raw values of a numeric attribute (e.g. age) are replaced by interval labels (e.g. 0-5, 6-14, 15-21, etc.) or conceptual labels (e.g youth, adult, senior) . The labels, in turn can be recursively organized into a concept hierarchy for the numeric attribute.

B. **Variable Creation**: It is a process to generate new features based on existing attributes. For an instance, date(dd-mm-yy) as an input variable in a data set. We can generate new variables like day, month, year, week, weekday that may have better relationship with target variable and highlight the hidden relationship in a variable. Some of the strategies for creating new features are:

1). Evaluation of errors generated by classifiers e.g. identify missing information and add new features to get the missing values and improve the classifier's performance[8].
2). Patel et al. proposed the construction of discriminating features by comparing labeled data with classifiers "confused region" with the opposite class.
3). Adjustment of feature weights in each iteration [9].
4). Crowd generating features [10].
5). Interactively debugging features etc. [11] .
6). Hsiang-Fu Yu et. al. used feature engineering approaches at two levels- Use of binarization and discretization techniques to create sparse feature sets and use of simple statistics on data to create condensed features respectively. Once the new features are constructed that better represent the underlying problem, can be fed into the classifiers to learn and improve the accuracy on unseen data.

Different feature Transformation and creation methods differ in the manner in which they are experimented and evaluated. Each transformed feature is obtained by evaluating some function over all the original's features.

IV. TOOLS/SYSTEMS FOR FEATURE ENGINEERING(FE)

Regrettably, FE for classification or any learning is a time consuming, tedious, often a trial and error process. There are some tools which accelerates FE process. In this subsection, we introduce few tools/systems that may be either promising or much needed for solving the feature engineering problems. The outstanding characteristic of these tools is to focus on the idea of extracting or transforming raw data into something which is more useful to the predictive model, rather than just searching for the best-known algorithm for the prediction rule. Some of the tools are inspected:

Turi: An environment to work with data and feature engineering function to design an effective machine learning models. FE tasks are grouped based on the features types. For an instance, numeric features- quadratic features, feature binning and numeric imputer, categorical features, text features, image features and other transformations like Hasher, Random projection, Transformer chain, Custom transformer etc. It performs number of feature transformations based on: Aggregate statistics (over various periods of time) of the raw input feature columns, items over various period of time (e.g. rate of change of aggregate usage) and user metadata (using the user_data parameter)

Azure Machie Learning Studio: A web based product, offers some FE functionalities. These are R script module to create features and Feature Hashing module for retrieving new generated feature. This FE process is followed by feature selection process for finally constructing the training data sets for predictive models. Analysts may create features for regression problems by executing R script module and also provides text mining by feature hashing module.

ZOMBIE: An IDE for feature engineers that drive FE through intelligently selection of inputs to model. It works on two stages: (a) Offline Indexing: grouping of similar raw feature set into index groups (b) Online querying: dynamically creates subset of data using index groups which is used to train the classifier. The IDE Basicallay does the following: apply user's functions or pre-written feature code on a large input sets mainly on web pages and evaluation of feature code's impact on any learning task.

FeatureFu: Its an open source toolkit consist of several libraries, tools and it is designed to perform advanced FE for an instance extended s-expression based feature transformation, derive features on top of others, or convert a light weighted model like logistical regression or decision tree into a feature, in an instinctive way without disturbing any code. It uses a java library "Expr" to evaluate mathematical s-expressions(parser and evaluator) written in Java and provides many feature contruction techniques like feature normalization, feature transformation, feature bining etc. Some transformations done by this tool kit functions are as: a) Feature normalization such as (min 1 (max (+ (* slope x) intercept) 0)) " - scale feature x with slope and intercept, and normalize to [0,1], here min and max are operators. b) Feature binding such as (- (log2 (+ 5 impressions)) (log2 (+ 1 clicks))) i.e combine no. of impression and no. of clicks. Nonlinear featurization: (if (> query _doc_ matches 0) 0 1) i.e negation of a query/document matching feature, also represent a small decision tree. c) Cascading modelling such as (sigmoid (+

(* x1 w1) w0)) i.e convert a simple logistic regression model into a feature, also Platt scaling for model score calibration.

COLUMBUS: A framework of declarative operations for FE and a novel optimizer that improves performance at scale. It is able to analyze large C/C++ projects. Its main functions are categorized into three types: a) Explore: use selection algorithms to select features by adding or removing features. b) Evaluate: Calulate numeric scores over feature sets eg. mean ,variance, correlation and covariance. c) Data-Transform: Does data manipulations over relational data sets eg. Squares, products of features.

DeepDive: A system to design features that assist in improving the knowledge base's accuracy. DeepDive's language inherits Markov Logic Networks' (MLN's) formal semantics. In particular, the system permits to see how output gets changed as the input changes. It deals with textual, structured and unstructured data, helps developers to think about features rather than algorithms and also to extract objects from the input documents like entitity, relation, a span of text called Mention, a phrase that connects two mentions called Relation Mention . It performs various functions such as a) Executes SQL queries and user defined functions to generate feasible mentions, entitities and relations. b) Collect relations from the existing database and use these examples to automatically generate the training data. c) Generates a factor graph, a model for learning and inferences.

FeatureSmith: A tool for constructing feature set without human intervention. This tool has been used to generate features for training machine learning classifiers to detect Android malware. It basically combines security knowledge described in natural language documents, such as papers published in security conferences and journals. Performs various functions such as a) Mine documents written in natural language(eg. scientific papers). b) Identify abstract behaviors associated with malwares. c) Represent and querying the knowledge about malware. d) Mapping of behaviours with concrete features.

FeatureForge: A tool for visually supported Feature Engineering and Corpus Revison. The tool has the capability to highlight problems with feature set and define new features to improve performance, finding objects which are not associated with any label/class and use those instances to update annotation guidelines by adding annotation rules and also assist in NLP to get comprehensive overview of the linguistic data and derive meaningful linguistic features. Besides these, shallow semantic parsing as well as for recognition of temporal expressions and also create interaction between feature representations and classification algorithms.

V. FEATURE ENGINEERING APPLICATION DOMAINS

The choice of feature engineering methods differs among various application areas. In the following subsections, we review different methods/techniques, adopted to show the effectiveness of feature engineering methods in classification problems. Nevertheless, the purpose of reviewing several papers is to show the effect of feature engineering methods in the classifier performance of discrete problems.

A. Text mining

Bag-of-words(count of words) is the most common way of representing a document in text mining. Scott[12], described new representation of text, based on Synonyms, hypernyms and and also combined different classifiers based on these different representations. Experiments were performed on each test set using 8 automatically-generated representations based on words, and phrases, synonym and hypernym relationships and evaluated feature values using RIPPER(Rule Based learner) and found significant improvement .

Gabor et. al. experimented[13] with different feature combinations in the SZTERGAK framework for keyphrase extraction to classify them as proper keyphrase category or not. It is a supervised keyphrase extractor system from scientific publications. For engineering features, first removed pertinent sentences based on their relative position in the doc. After that, retrieve candidate based on n-grams(up to size 4 grams) confined by POS patterns then ranked candidates based on word length, POS patterns. The different types of features dealing with keyphrase extraction are: the document level features, the corpus level features, The extended feature set and external knowledge based features. The results were obtained with different features combination and analysed with respect to F-score and accuracy. With different feature combination or transformation significant improved has been achieved.

Cramel et.al. noticed the effect of feature engineering for the problem of rainfall prediction on 21 different data sets of cities across Europe. It was a previous work extension carried out on Genetic programming(GP). It is further extending with extra set of features using moving average, standard deviation, adjacent difference, Time last min-max, magnitude of min-max, min-max last contract length etc to assist GP. and seen the positive effect on genetic programming. The authors note that the GP outperformed with extra features than its original features. [14]

Researchers have investigated the impacts of different sets of features such as tagging scheme for training a CRF-based timex recognition system, head words, phrase tag, and addresses the different aspect of improving the overall performance [15].

B. Link Prediction Applications

Link prediction is an important network science problem in many domains such as social networks, chem/bio-informatics, etc. Due to the dynamic nature of networks, time information gives better performance in supervised link prediction. They proposed two steps feature construction with time feature matrix. The idea is to: 1) Calculation of feature matrix with respect to each time unit. 2) Compute meta features of the step 1. Features by slicing out the time column. Some features wer computed using Linear Algebra Technique, Common Neighbours, Preferential Attachment, Adamic-Adar measure, Common Keywords, Degree mixing probability etc. Feature vectors with time information were tested for link prediction and achieved the better classification performance.[16]

C. Applications of Analysing Online Behaviour

Researchers explored the procedure of engineering features for constructing models that improve learners understanding of online behavior in MOOCs and evaluated different feature set to achieve the accurate prediction by using them a supervised learning scenario. They proposed some techniques such as Brainstorming feature ideas, Crowd sourcing, Operationalizing features ideas/proposals, Randomized Logistic Regression Analysis, Linked information from multiple sources or MOOCdb etc. Diffetent feature sets were evaluated and examined in predictive accuracy[17].

D. Knowledge Base Construction(KBC) Applications

KBC is the process of dynamically populating Populating RDBMS with information from different sources(unstructured data sources. To construct KBC, authors focused on feature engineering. They used DeepDive system based on markov Logic for constructing new features. Its execution consist of grounding and statistical inference and learning. Different approches are described in terms of performing feature engineering rather than traditonal approaches to speed up the creation of KBC Systems such as set of random variables defined to create new query relations where each instance relate to one random variable after that identified the relation among them then estimated the weights, performs inferences and produce the output databases. Discussed various probabilistic and machine learning techniques to engineer a better system[18]

E. Health Related Social Networks

Adverse Drug Reaction(ADR) is one of the concern which has drawn more attention from the public as it leads to harness public health and economic loss to our society as well. Researchers developed, an ADR event monitoring system, identifies ADRs related concepts from twitter post Researchers has studied several techniques and discussed in their work. Some of the methods they used such as Twokenizer, Hunspell, the numerical normalization and stemming approach, sequential labeling using IOBES scheme. Systems were evaluated using two paired criteria, Precision(p)=TP/(TP+FP) and Recall(R)= TP/(TP+FN) and the compined criterion, F-measure(F)= 2XPXR/(P+R), the notations of TP, FP and FN stand for true positives, false positives and false negatives, respectively.Results were evaluated and demonstrated, several feature combination sets could better classify texts or posts that are decisive of ADRs[19].

F. Fraud Detection Application

Authors studied and expanded the transaction aggregation strategy, and proposed construction of new set of features by analysing the transaction time behaviour over period that uses the von Mises distribution for Credit Card Fraud Detection Problem.(CCFD) Some comparisons were performed on different sets of feature by using classification algorithms; cost Insensitive and example dependent cost-sensitive techniques. They analyzed an indiviual card holders behaviour while constructing CCFD model and found that feature engineering increases the performance by 200% compared to using only the raw transaction data[20].

Though, Feature engineering is time consuming, as it is basically about transforming training data and strengthen it with additional features in order to make learning algorithms more effective. Data scientist considers feature engineering as one of the crucial part of learning process as it heavily relies on creativity and human intervention[21]. There are many other techniques to create features like helmert coding, forward difference encoding, canonical correlation, Yeo-Johnson transformation etc. However, based on the underlying problem, additional features are created or transformed as well as it also depends on the chosen model that will hold through all the features without being slow.

VI. CONCLUSION

Building an efficient classification model for classification problems with some additional relevant features along with different data sets and different sample size is important. The main tasks are construction of additional relevant features from the existing feature sets, and then feature selection to discard redundant, irrelevant or highly correlated features. Selected features are minimal set of independent features which explain the patterns in the data and then classified outcomes successfully. In this survey paper, we presented an overview on feature engineering tools and techniques that make it possible for analytics to build classifiers more effectively. Experimental results in various research shows that feature engineering simplified feature selection process and the total number of parameters needed effectively, thereby obtaining a higher classification accuracy compared to non-engineered features. Effectiveness of a model can be determined by iteratively inputting different combinational features sets into a given model but the real challenge is the evaluation of inputs that fed into the given model.

REFERENCES

[1] M. Anderson, D. Antenucci, V. Bittorf, M. Burgess, M. J. Cafarella, A. Kumar, F. Niu, Y. Park, C. Ré, and C. Zhang, “Brainwash: A Data System for Feature Engineering.,” in Proc. CIDR 2013, 2013.

[2] P. Domingos, “A few useful things to know about machine learning,” Commun. ACM, vol. 55, no. 10, pp. 78–87, 2012.

[3] Isabell Guyon and André Elisseeff, 2006. “An Introduction to Feature Extraction,” in Guyon, Isabelle, Steve Gunn, Masoud Nikravesh, and Lofti A. Zadeh, eds. Feature Extraction: Foundations and Applications, pp. 1-25. Springer Berlin Heidelberg, 2006.

[4] J. Cheng and M. S. Bernstein, “Flock: Hybrid Crowd-Machine Learning Classifiers,” 2015, pp. 600–611.

[5] P. Domingos, “A few useful things to know about machine learning,” Commun. ACM, vol. 55, no. 10, pp. 78–87, 2012.

[6] https://en.wikipedia.org/wiki/feature-engineering.

[7] What is the intuitive explanation of feature engineering in ML?-Quora, www.quora.com. Retrieved, 2015-11-11.

[8] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python. O’Reilly Media, Inc., 2009.

[9] H. Raghavan, O. Madani, and R. Jones, “InterActive Feature Selection,” in Proc. IJCAI 2005, 2005, vol. 5

[10] J. Cheng and M. S. Bernstein, “Flock: Hybrid Crowd-Machine Learning Classifiers,” 2015, pp. 600–611

[11] F. Heimerl, C. Jochim, S. Koch, and T. Ertl, “FeatureForge: A Novel Tool for Visually Supported Feature Engineering and Corpus Revision,” in Proceedings of COLING 2012: Posters, Mumbai, India, 2012, pp. 461–470.

[12] Scott Sam, Matwin Stan, Feature Engineering for Text Classification, Proceedings of ICML-99, 16th International Conference on Machine Learning,1999

[13] Gabor Berand, Richard Farkas, SZTERGAK: Feature Engineering for Keyphrase Extraction, Proceedings of the 5th International Workshop on Semantic Evaluation, p. 186-189, July 15-16, 2010, Los Angeles, California.

[14] Cramer, S., Kampouridis, M., Freitas, A.A., Feature Engineering for Improving Financial Derivatives-based Rainfall Prediction, IEEE World Congress on Computational Intelligence, Vancouver, Canada (2016).

[15] F. Adafre and M. de Rijke. 2005. Feature engineering and post-processing for temporal expression recognition using conditional random fields. In Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing.

[16] Jeyanthi Narasimhan, Lawrence Holder, Feature Engineering for Supervised Link Prediction on Dynamic Social Networks, School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164-2752, USA, 7 Oct. 2014.

[17] Kalyan Veeramachaneni , Una-May O’Reilly , Colin Taylor Towards Feature Engineering at Scale for data from Massive open Online Courses, arXiv:1407.5238v1 [cs.CY] 20 Jul 2014.

[18] Christopher Re, Amir Abbas Sadhgein, Zifei Shan, Jaeho Shin, Feiran Wang, Sen Wu, Ce Zhang, Feature Engineering for knowledge Base construction(KBC), Copyright 2014 IEEE.

[19] Dai, H.J., Touray, M., Jonnagaddala, J., Syed-Abdul, S.: Feature engineering for recognizing adverse drug reactions from Twitter posts. Information **7**(2), 27 (2016).

[20] Bahnsen A. C., et al. (2016). Feature engineering strategies for credit card fraud detection. Expert Systems with Applications, Vol. 51, pp. 134-142.

[21] Wang Christopher Alex, Feature Factory: A Collaborative, Crowd-Sourced Machine learning System, Massachusetts Institute of Technology 2015.