CLASSIFICATION OF FOLKTALES BASED ON DIETARY PREFERENCES

MASTER'S THESIS IN LANGUAGE SCIENCE TECHNOLOGY

By:

Erdenesuvd Dashjamts

Advisors:

Prof. Josef VAN GENABITH

Thierry DECLERCK

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

Darkhan, Sep 28, 2017

Abstract

During the last decade, particularly after 2010, there has been a sharp increase in the number of people who chose a meatless lifestyle. This trend is continuously growing rapidly around the world resulting in a higher demand of children's books for vegetarian families. Many authors already created numerous books in this context; however, we wonder if traditional folktales would suit this need. In this thesis, we took advantage of computational advancement to classify folktales written in or translated into English around the 1900s from different cultures and locations into five different dietary classes. For classification, rule-based and hybrid machine learning systems were implemented. Due to the imbalanced nature of the small corpus we created, the method of oversampling with virtual examples was used to augment and balance out our dataset for the latter algorithm and both models achieved fairly similar results. The final application is deployed online aiming to assist anyone who loves traditional fairy tales to find out if the story they are intending to read suits their dietary preferences. In the future, one could extend the system to be able to process more specific contents.

Keywords: folktale, fairy tale, classification, support vector machine, vegetarian, vegan, fruitarian, oversampling, rule-based, virtual examples

Acknowledgement

I thank my supervisors, professor Les Sikos and everyone else in the LCT program for helping make this thesis a reality.

Table of Contents

List of Figures

List of Tables

1 Introduction

Palm-sized computers, robot workers, changing environment… not many things could surprise you these day. Being a vegetarian, a vegan, or even a fruitarian is one of these things. It is no longer as surprising as it was fifty years ago; it is no longer as surprising as it was five years ago; not even one year ago. In fact, vegetarianism is on the rapid rise on our planet due to specific reasons such as ethical, health and personal although it is already an old tradition in a few countries like India. Young adults, children, millenials and whole families are choosing this lifestyle consciously in traditionally meat-consuming countries; as a result, there is a necessity for those parents to select the books to read to their children carefully. The stories should suit their way of living. There are many modern fairy tale books written by current authors specifically for these concerned mothers and fathers. Nevertheless, people still love the good old folktales that have been circulating around for many many years. This thesis aims to help parents or anyone who loves reading traditional folktales to be confident about which folktales suit their eating habits.

Thanks to the advancement in computing technology in the recent years, we are now able to classify texts, images and sounds into different categories. We can analyze over a given text and parse its structure in different ways. Therefore, we will try to take the advantage of the up-to-date computational algorithms to extract vital information from the folktales and then classify over the modified data.

We will classify a selection of folktales from more than seven different cultures and locations, from the late 1800s till the early 1900s, all translated into English.

If you have any suggestion for further improvement, please let us know.

1.1 *Research Question*

This thesis will try to answer the question of whether we could classify the folktales computationally according to dietary habits and so those classified traditional folktales could be pre-recognized as whether they could be suitable for the increasing number of vegetarian humans.

1.2 *Research Motivation*

This section explains why we need vegetarian friendly books and stories.

According to the Vegetarian Resource Group, the amount of vegetarians comprised only 1% of the U.S. population in 1997[1] and it went up to 3% in 2009 which almost doubled to 5% (16 million people) in 2011[2] and about 33% of the Americans are consuming vegetarian and vegan meals more frequently[2]. A similar phenomenon is observed throughout the world. In New Zealand, where meat is consumed heavily, Roy Morgan Research has found that the percentage of people who prefer vegetarian diet permanently or mostly grew by 27% since 2011[3]. The number of vegans in the U.K. has gone up by 360%, from 150000 to 542000, in the last decade since 2006 according the Vegan Society[4]. The amount is highest in Germany in with as high as 10% of the total population choosing a meatless lifestyle, making the country the most vegetarian among its counterpart European neighbors[5]. Germany's environment minister Barbara Hendricks also announced that no more meat products to be served at official functions of the ministry starting from January, 2017[6]. This phenomenon coincides with Google Trend search statistics as well, shown in Figure 1 and Figure 2.

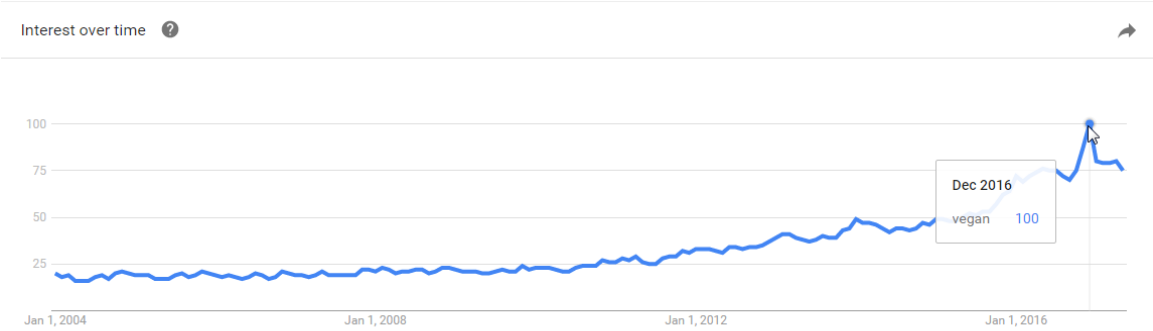

Figure 1. Search trend for the term "vegan" since 2004

As can be seen from the graph, the *vegan* search spiked during the Christmas and New Year's time of 2016 due to perhaps the holiday season nature such as having vegan guests over or making a major resolution for the coming year. It was still maintaining high index as it never has been ever since 2004 and we expect it to continue in the future.

---

[1] https://www.vrg.org/journal/vj97sep/979poll.htm
[2] http://www.vrg.org/blog/2011/12/05/how-many-adults-are-vegan-in-the-u-s/
[3] http://www.roymorgan.com/findings/6663-vegetarians-on-the-rise-in-new-zealand-june-2015-201602080028
[4] https://www.vegansociety.com/whats-new/news/find-out-how-many-vegans-are-great-britain
[5] http://www.vrg.org/blog/2016/09/26/vegetarian-market-in-germany/
[6] http://www.spiegel.de/politik/deutschland/umweltministerium-serviert-bei-veranstaltungen-nur-noch-vegetarische-kost-a-1135231.html

Figure 2. Search trend for the term "vegan" by region[7]

The most *vegan-searching* country is Australia, followed by Canada, the United States, New Zealand and Germany, which is paralleled with our survey results for the respective countries.

Interestingly, this actuality creates a new inquiry in many different lifestyle directions, including vegetarian-vegan friendly story books for children growing up in this way. Certain websites started releasing a list of vegetarian-vegan friendly books for young readers and on-the-rise parents too[8]. Out of curiosity, I chose a well-known online book site, goodreads.com, to compile the dates when these kinds of books were published. As a result, out of total 86 vegetarian-vegan friendly books for kids on goodreads[9], 14 (16.3%) were published before 2000, 32 (37.2%) books were published between 2000 and 2010, 40 (46.5%) books were published within the last few years. This shows there has been an enlarged demand for this type of books for children and the rate of acceleration somewhat coincides with the diet practice trends in the graphs.

We can clearly see that there is a suggestion of choosing life-style friendly stories from the above information and this can not only be applied to modern authored stories but also to those that were written in earlier years too.

We can intuitively remember that many early and even more adapted versions of folktales involved eating various types of flesh in ways that would make modern vegetarians shudder. Therefore our work could help those on the trendy diet to enjoy traditional tales.

1.3 *Research Aim*

The aim of this research project, therefore, whether we are able to classify folktales into different dietary categories computationally so that we can find out if most or some of those tales could be read by and suitable for the new generation of the emerging vegetarian societies.

---

[7] The adjusted relative scores are based on the relative popularity of the term "vegan" in each country within the specified time range. The country with the highest relative interest has a score of 100.

[8] https://www.youngveggie.org/document.doc?id=461, https://www.vrg.org/family/Vegetarian-friendly_Kids_Booklist.pdf

[9] http://www.goodreads.com/list/show/3838.Vegetarian_Vegan_Friendly_Books_for_Kids

Just as a side check, it would also be interesting to find out how certain cultures dealt with different categories of food in our random selection of folktales. This is a question that needs more thorough investigation in further studies.

2 The Conceptual background

2.1 *Folktales and Fairy Tales*

A folktale is an oral narrative which circulates between (groups of) people for longer or shorter time. Folktales are part of 'oral art' (verbal art) of daily life, past and present. [10]

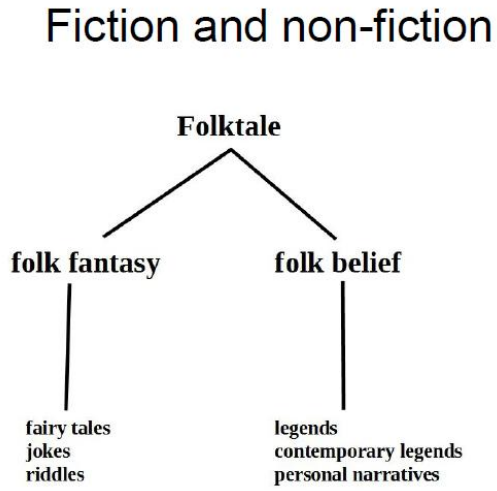

Figure 3. Folktale genre classification[11].

From Figure 3, we could conclude that fairy tales are a kind of folktale. The word *fairy tale* was translated from the French term *conte de fées*. Fairy tales are understood as fantasy and not fixed in time and place. Generally, the protagonist starts an adventure to solve a problem and gets magical help from others, including animals. Antagonists may usually have supernatural power. And eventually, the hero(ine) wins by being kind, brave, smart and lucky as a reward. Most of them end happily ever after. And fairy tales are longer than other subgenres of folktales. The Brothers Grimm and Vladimir Propp are well-known for their effort to study fairy tales scientifically.[12]

2.2 *Folktale Classification*

Traditionally, folktales are known to be classified using two basic systems. The content-focused folkloristic approach classifies using either a more general and recurring theme based type index (ATU: Aarne/ Thompson/ Uther) or a more detailed and recurring element based motif index (TMI: Thompson motif-index) both of which complement and combine with

[10] Meder, Theo. (2017). Dutch Folktales: Classifications [Lecture slides#3]. University of Groningen. Retrieved from https://nestor.rug.nl/.
[11] Meder, Theo. (2017). Dutch Folktales: Classifications [Lecture slides#4]. University of Groningen. Retrieved from https://nestor.rug.nl/.
[12] Meder, Theo. (2017). Dutch Folktales: Fairytales [Lecture slides]. University of Groningen. Retrieved from https://nestor.rug.nl/.

each other.[13] The structuralistic way, suggested by Vladimir Propp in 1928, focuses more on the function motifs of folktales and explores 31 functions and 7 different characters.[14] There are some folklorists who tried to classify specific group of tales using the extended version of the traditional index-based classification.[15]

Nowadays, computing technology has reached an advanced level which allows us to use its power to classify folktales in whatever way we want. Although there have been certain ongoing attempts in folktale classification using the computational advancement, much more needs to be done in this discipline. Currently, a number of researchers are trying to classify folktales and other types of folklore materials using computational methods. One such project is Tunes & Tales[16] by the Meertens Institute and other related universities. Under the umbrella of this project, there was a study into generating a genre-based classification[17]. The project aimed to implement automatic classifications in folktales and folksongs. Another research by researchers from various Dutch institutes, Folktale as Classifiable Texts (FACT), employed machine learning clustering systems for classification to discover folktale specific properties that human annotators might miss upon.[18]Other classification experiments using the traditional type and motif indices were implemented at the Meertens Institute in 2016 [19,20]. There was an experiment focused on a ranked list of multiple possible labels rather than one story – one label way[21]. In another paper, a genre-based classification through computational way is suggested[22]. Researchers from Germany worked on creating linked ontological representation of the past classification systems such as ATU and TMI[23]. Outside of the scope of these few projects, it was nearly impossible to find other research studies trying to solve folktale classifications computationally. Other works that are not really classification solutions but computational solutions towards the problems that may help story classifications include sentiment analysis on 453 fairy tales from the Fairy Tale Corpus[24], automatic annotation of characters' emotions in stories[25] and finding structural similarities in narrative texts[26].

---

[13] Harun, Harryizman, and Zulikha Jamaludin. "Folktale conceptual model based on folktale classification system of type, motif, and function." *Proceeding of the 4th international conference on computing and informatics (ICOCI)*. 2013.

[14] Propp, Vladimir. "Morphology of the Folktale, trans." *Louis Wagner, 2d. ed.* (1968).

[15] For instance: Seki, Keigo. "Types of Japanese folktales." *Asian Folklore Studies* 25 (1966): 1-220.

[16] http://www.ehumanities.nl/computational-humanities/tunes-tales/

[17] Nguyen, Dong-Phuong, et al. "Automatic classification of folk narrative genres." (2012).

[18] https://www.utwente.nl/en/eemcs/db/research/currentprojects/fact/

[19] Meder, Theo, et al. "Automatic enrichment and classification of Folktales in the Dutch Folktale Database." *Journal of American Folklore* 129.511 (2016): 78-96.

[20] Nguyen, Dong, Dolf Trieschnigg, and Mariët Theune. "Folktale classification using learning to rank." European Conference on Information Retrieval. Springer Berlin Heidelberg, 2013.

[21] Broadwell, Peter M., David Mimno, Timothy R. Tangherlini. "The Tell-Tale Hat: Surfacing the Uncertainty in Folklore Classification." Journal of Cultural Analytics (2017).

[22] Harikrishna, D. M., and K. Sreenivasa Rao. "Children story classification based on structure of the story." Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on. IEEE, 2015.

[23] Declerck, Thierry, and Lisa Schäfer. "Porting past Classification Schemes for Narratives to a Linked Data Framework." Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage. ACM, 2017.

[24] Mohammad, Saif M. "From once upon a time to happily ever after: Tracking emotions in mail and books." Decision Support Systems 53.4 (2012): 730-741.

[25] Lombardo, Vincenzo, et al. "Automatic annotation of characters' emotions in stories." International Conference on Interactive Digital Storytelling. Springer, Cham, 2015.

[26] Reiter, Nils. Discovering Structural Similarities in Narrative Texts using Event Alignment Algorithms. Diss. 2014.

This project intends to find a way to classify folktales according to human dietary styles. All previous folktale classifications had a particular focus to organize folktales; however, as far as we are concerned, not one of them fixated on foods. On another note, whenever any food was mentioned in a paper related to folktales or fairytales, it was usually related to origin[27] or symbolism[28,29,30,31] of foods. Therefore, our attempt of classifying folktales regarding different food categories could start an opening for future studies in this direction.

2.2.1 Computational Classification

Classifying data computationally has provided us a possibility to categorize massive amounts of data within a relatively short time frame with less human caused errors. The machine-based computational classification can help us in two major ways. They should either follow the rules written by us (the Rule-based Classification) or learn by themselves from training data (Machine Learning). Like most everything in life, however, the machine learning has its own disadvantages; it can only approximate classification at a high rate given good and quality data resembling the natural variations of the real task so that the machines can learn well from them. In our problem, the nature of our five dietary classes and the class specific terms, which are nested inside one another like the Russian doll *Matryoshka*, produces a hierarchical decision structure where the classification decision is based on the presence of the food related terms belonging to the widest-scope class in the tale (See Table 1). Any hierarchical decision structures can be solved with both of the computerized classification methods, not only through a rule-based approach but also using the self-learning machine algorithm. We will employ the hybridized version of the latter to achieve the best results.

2.2.1.1 Rule-based Approach

When we try to classify text documents using the power of computers, we write an algorithm telling the computer each and every single detail satisfying specific conditions. For that, we have to create sets of IF-THEN rules generalized over the corpus to give the correct label to the text. For example:

if eatTerm==”eat” AND foodTerm==” apple”: then class= “fruitarian”

Intuitively, this method can be useful for domain specific classification problems (folktales in our case) with sparse data.

2.2.1.2 Hybrid Machine Learning Approach

The modern machine learning algorithms have the ability to learn from the given data by themselves and make decisions with or without detailed programming rules. This method of classification recognizes and learns from certain patterns from the training part of a dataset

---

[27] Roosman, Raden S. "Coconut, breadfruit and taro in Pacific oral literature." The Journal of the Polynesian Society 79.2 (1970): 219-232.

[28] Honeyman, Susan. "Gingerbread Wishes and Candy (land) Dreams: The Lure of Food in Cautionary Tales of Consumpion." Marvels & Tales 21.2 (2007): 195-215.

[29] Andrievskikh, Natalia. "Food Symbolism, Sexuality, and Gender Identity in Fairy Tales and Modern Women's Bestsellers." Studies in Popular Culture 37.1 (2014): 137-153.

[30] Canonici, Noverino Noemio. "Food in Zulu folktales." Southern African Journal for Folklore Studies 2.1 (1991): 24-36.

[31] Flanagan, Michael. "Cowpie, Gruel and Midnight Feasts: the representation of Food in Popular Children's Literature." (2012).

(e.g., corpus of folktale documents) and then predicts and identifies the corresponding categories (e.g., set of dietary categories) for the testing part.[32] Possible patterns in text classification could be a set of category specific terms. For example: {“avocado”, ”mango”, ”nut”, “grape”, “orange”} for the category of *fruitarian.* A folktales containing those terms should be classified as a *fruitarian* tale. A folktale containing a set {“beef”, ”eat”, ”chicken”, “devour”, “horse”, “apple”, “potato”} could be classified as an *omnivorous* tale.

In our project, the text features are a bit different than those in usual text classification problems. Each folktale contains mostly unrelated noisy terms with only a few or no food related terms and this makes the machine totally confused. Therefore we had to recreate micro versions of the tale documents (See Appendix D: Feature Extraction) consisting of only related terms so it becomes easier for the machines to learn. This feature extraction process requires certain handwritten rules and that makes our machine classification hybrid.

2.3 *Classification Categories*

2.3.1 Food classes

Remember the pumpkin coach trick from Cinderella, or the gingerbread house, or the poisoned apple from Snow White? Undoubtedly, many fairy tales contain food related context and some foods made them be remembered distinctively. We will classify them depending on what kind of foods are enclosed in each one of them. For that reason, we need definite classes where the tales could belong to.

There are a number of different diets practiced depending on various factors[33]. The majority of people around the world are omnivore eaters, meaning that they would eat many different food items such as meats, eggs, dairies and so on without much restriction. However, many other humans follow a meatless lifestyle which is also hierarchically subcategorized into even stricter and distinct subtypes of vegetarianism[34,35]. We could sum them up into three sub classes, namely, vegetarian, vegan and fruitarian. Now we have four different dietary sub categories as omnivorous, vegetarian, vegan and fruitarian. However, there could also be folktales which do not contain any food related terms and they could be considered as neutral. Therefore, our dietary categories in this thesis are as following: fruitarian, vegan, vegetarian, omnivorous and neutral.

The five dietary categories, namely, omnivorous, vegetarian, vegan, fruitarian and neutral, will serve as our distinct dietary classes for our classification problem.

2.3.2 Food groups

To make up dietary classes, we need different food groups to decide which food item belongs to which food group and eventually decide on which food group belongs to which food class.

[32] https://en.wikipedia.org/wiki/Statistical_classification
[33] https://en.wikipedia.org/wiki/List_of_diets
[34] https://en.wikipedia.org/wiki/Vegetarianism
[35] Piper, Brenda. Diet and nutrition: a guide for students and practitioners. Springer, 2013.

Various information sources about health and food recommend main food groups and we cite some of them below.

MyPlate is a healthy eating style food guidance from the United States Department of Agriculture introduced in 2011 with latest updates in January 25, 2017. According to this guidance[36], there are five main food groups, namely, fruits, vegetables, grains, dairy and protein (meat, poultry or seafood, legumes, eggs, nuts and seeds) and, in addition, oils.(See Figure 4)

The Vegetarian Society of the United Kingdom 'Eatwell Plate'[37] suggests similar food groups too: Fruits and vegetables, grains and starchy foods, dairy, fatty sugary foods, meat replacing foods (eggs, legumes, nuts and seeds). (See Figure 5)

Physicians Committee for Responsible Medicine developed the Four Food Groups[38] in 1991 for a healthy vegan lifestyle and suggests daily consumption as its Power Plate: Fruit, Legumes, Whole Grains and Vegetables. (See Figure 6)

Figure 4. Myplate by the United States Department of Agriculture

Figure 5. Eatwell plate by The Vegetarian Society of the United Kingdom

Figure 6. Power plate by Physicians Committee for Responsible Medicine

Vegetarian Nutrition developed a Food Pyramid[39] with groups: Whole Grains, Legumes and Soy, Vegetables, Fruits, Nuts and Seeds, Vegetable oils, Dairy, Eggs and Sweets. According to International Vegan Association[40], the food groups are Legumes Nuts and Seeds, Grains, Vegetables, Fruits.

All in all, based on the above guidelines we could sum up the total food groups as following:

Fruits (including Nuts and Seeds), Vegetables, Grains (mostly wheat and rice products), Legumes (mostly beans, lentils and peas), Meat including Poultry and Seafood, Dairy and Eggs.

[36] https://www.choosemyplate.gov/protein-foods
[37] https://www.vegsoc.org/eatwellplate
[38] http://www.pcrm.org/health/diets/vsk/vegetarian-starter-kit-new-four-food-groups
[39] http://www.vegetariannutrition.org/food-pyramid.pdf
[40] http://www.internationalvegan.org/nutrition/

Now we need to allocate these groups to their respective food classes. The Omnivores eat all types of foods and therefore all types of meats and flesh distinguish this class from the other classes. Vegetarians would include dairy and sometimes eggs and even fish because the term is loosely utilized in this context; however, in this thesis, the fish is considered as meat and the eggs and dairy would belong to the vegetarian class. Vegans would consume only plant based foods and fruitarians would consume only or mostly fruits of plants.

In summary, we consider that fruitarian diet consists of all types of fruits and nuts and seeds, vegan diet adds all kinds of vegetables and grains and legumes on top of fruitarian diet, vegetarian diet adds dairy and eggs, and lastly, the omnivore diet includes all types of meats including fish on top of the vegetarian diet.

neutral	-
fruitarian	Fruits, nuts, seeds
vegan	Fruits, nuts, seeds, vegetables, grains, legumes
vegetarian	Fruits, nuts, seeds, vegetables, grains, legumes, dairy, eggs, honey
omnivorous	Fruits, nuts, seeds, vegetables, grains, legumes, dairy, eggs, honey, meat

Table 1. Food groups per food class

3 Vocabulary and Data

Having good data is more important than or at least as equal as to having the best classifier. Even the best classification algorithms cannot perform well without quality training data that can represent the real population as close as possible and are sufficiently large to capture most of the variations in the real data as well.

3.1 *Food Vocabulary*

We intuitively know that folktales contain simple and general words. And the same principle applies in terms of food related terms. With these in mind, initially, we tried creating our food item terms for each food group from WordNet[41]. However, the synonyms were too few and the hyponyms were mostly rare, wrong or unrelated, meaning the vocabulary we collected from WordNet was not simple or suitable enough for folktales. For example:

Synonyms[42]*:* Synonyms for the term *fruit* resulted in only two terms: *fruit* and *yield*.

Hyponyms[43]: Most of the hyponyms were not the general type terms such as *apple* that are used in folktales mostly for the following reasons:

Non-general term: May apple instead of the general type term *apple*; *wild cherry* or *chokecherry* instead of simply *cherry* as always in folktales.

[41] https://wordnet.princeton.edu/

[42] https://en.wikipedia.org/wiki/WordNet: Words from the same lexical category that are roughly synonymous are grouped into synsets. For example: *apple* is synonym of *pear*

[43] https://en.wikipedia.org/wiki/WordNet: Y is a hyponym of X if every Y is a (kind of) X (apple is a hyponym of fruit)

Ambiguous and uncommon: certain scientific or botanical type of fruit which are not suitable in literature: *ear*

Unrelated: accessory fruit

Rare: marasca

Unnecessary repetition: rose hip, rosehip

When we calculated the term frequency for the general term *apple*, which was not found among the results from the *fruit* hyponyms from WordNet; there were altogether 178 occurences in our corpus of 804 tales while there was 0 occurrence for terms such as *May apple* and *marasca*.

As a result, only 3 out of total 44 hyponyms for the term *fruit* were the real general fruit names that usually occur in folktales, namely, *olive*, *acorn* and *berry*. The other 41 were either strange, rare, or unrelated items. Similar frequency phenomenon was observed for the term *edible fruit*. Only 35 out of total 128 fruit names were useful for folktales although it contained general terms such as *apple* and *berry*.

Synonyms for *eat* would return only *feed*, *consume* and *corrode*, which is never used as food term in folktales. Hyponyms included unnecessary words such as *bolt*, *dip*, and *dunk* and 9/38 verbs were usable. With *usable*, we mean *occurs in folktales.*

Therefore we decided to create our own list of common food related terms based on the vocabulary from different sources such as a reliable website[44], the folktales during the annotation process and supervisor's recommendation etc[45]. These food items would form the food groups (as shown in Table 1) and eventually each food class by merging. The resulting lists are as below and we address these terms as *food terms* in this thesis.

(Note: We divided the corpus into a training and a test sets and reserved the test set separately as unseen documents for our final evaluation. Therefore the terms displayed below are either generic food terms pulled from reliable web sources or the terms obtained from the annotation. The generic food terms can also be used to classify the test sets.)

[44] http://vegetablesfruitsgrains.com

[45] Although this approach would include the general terms from web and other sources and specific terms from the corpus itself, it would still lack in unseen terms from unseen folktales.

Class	Terms	Count
Fruit	almond, apple, apricot, avocado, banana, barberry, berry, bilberry, blackberry, blackcurrant, blueberry, boysenberry, breadfruit, carob, cherry, cloudberry, cocoanut[46], coconut, cranberry, cucumber, date, dragon fruit, durian, elderberry, fig, fruit, goji berry, gooseberry, gourd, grape, grapefruit, guava, huckleberry, jackfruit, jujube, kiwifruit, kumquat, lemon, lime, lingonberry, loganberry, longan, loquat, lucuma, lychee, mango, mangosteen, marionberry, melon, mulberry, nectarine, noni, nut, olive, orange, papaya, passion fruit, peach, pear, persimmon, pineapple, pitaya, plantain, plum, pomegranate, pomelo, quince, raisin, rambutan, raspberry, redcurrant, rhubarb, salmonberry, starfruit, strawberry, tamarillo, tamarind, tangelo, tangerine, tomato, walnut, watermelon	99
Vegetable	artichoke, arugula, asparagus, aubergine, beet, bell pepper, bok choy, broccoli, cabbage, calabash, capers, carrot, cassava, cauliflower, celery, celtuce, chard, chayote, daikon, edamame, eggplant, endive, fennel, fiddlehead, galangal, garlic, ginger, horseradish, kale, leeks, lemongrass, lettuce, maize, mushroom, nopale, okra, onion, parsley, parsnip, pepper, plantain, potato, pumpkin, purslane, radicchio, radish, rampion, rutabaga, seaweed, shallot, spinach, sprout, squash, sweet potato, taro, tomatillo, tuber, turnip, vegetable, water chestnut, water spinach, watercress, yam, zucchini	71
Grain	amaranth, barley, bread[47], buckwheat, cake[48], chapatti, confectionery, cookie, corn, couscous, crumb, dough, durrie, flour, gingerbread, grain, loaf, maize, millet, mudki, oat, paddy, pancake, pastry, quinoa, rice, rye, sorghum, spelt, sugarcane, sweetmeat, teff, triticale, wheat	37
Legume	bean, chickpea, chocolate, lentil, pea, peanut, pulse	7
Dairy	butter, cheese, curd, custard, ghee[49], ghi, milk, porridge, quark, whey, yoghurt, yogurt	11
Honey	Honey	1
Egg	Egg	1
Flesh unambiguous	bacon, beef, venison, haggis, meat, mutton, pork, sausage, steak, stew	9
Flesh ambiguous	animal, ant, antelope, attendant, beetle, bird, boar, body, bone, boy, brother, buffalo, cadaver, capon, carcase, carcass, cat, cattle, cavalcade, chicken, child, companion, corpse, cow, creature, crow, daughter, deer, dog, elephant, enemy, entrail, family, father, fish, flesh, fly, fowl, frog, game, girl, goat, goose, grandfather, grandmother, grasshopper, grub, hare, heart, hog, horse, human being, insect, kidney, king, lamb, lambikin, limb, liver, lung, man, member, merchant, mice, minister, mother, offspring, organ, owl, ox, parent, part, person, pig, pigeon, piglet, prey, prince, queen, rabbit, raja, raven, rooster, servant, sheep, sister, snake, son, stomach, subject, turkey, worm	91
Total	Specific food terms from the training: The underlined terms are found either during the tale annotation or unusual terms used in specific tales. This is obtained from only the training set and can be used for both the training and test sets. General food terms: The terms that are not underlined are either from certain reliable websites or from common everyday foods. They can be used for both training and test sets. Specific food terms from the test set: In addition to the General food terms, there were certain test set specific food terms that were used to augment the test set tales to balance out. These are unseen features to our classifiers. Devourer: ant-eater, likho[50]; Ambiguous flesh terms: relative, fellow, rat, paunch; Dairy: kasha; Vegetable: kissel; Fruit: chestnut, citron	327

Table 2. Vocabulary list for *food terms*

[46] A variant of spelling for "coconut" that are found in some tales.

[47] The word *bread* in bread related expressions such as earn/ win/ beg +possessive adjectivial pronouns/ word *daily*+bread would not considered as a food item in our study

[48] in the old form, *cake* means simple bread baked from both sides, so it's vegan, not complicated recipes including eggs

[49] A variant of spelling for "ghee" that are found in some tales.

[50] A type of monster with one eye; it usually appears in Slavic tales

For food related activities, we use the eat-related terms as in the following table and we will address them as *eat terms* in this thesis.

Eat	bake, boil, cook, devour, eat, fry, gobble, roast, salt, stew, swallow	18

Table 3. Vocabulary list for *eat verbs*

For obvious flesh eaters, we collected the following terms in Table 4 and we address them as *devourer*.

Devourer	cannibal, man-devourer, man-eater, man-eating, monster, oger, ogre, rakhas, rakhasa, rakshas, rakshasa, rakshasi, raw-eater	12

Table 4. Vocabulary list for *devourers*

These lists are meant to be expanded and improved constantly for the training portion of the data as the corpus size increases so that the classifiers would struggle less with unseen test features. The machine learning algorithm may learn how to classify without the vocabulary created given enough training samples. Currently, we need the vocabulary for both types of classifiers. A similar approach of compiling food vocabulary based on the lists from two related websites was implemented to determine healthy and unhealthy foods mentioned in tweets from three American cities[51].

3.1.1 Food Vocabulary to Ignore

We decided not to include any of the general food items that would not convey any class specification.

3.1.1.1 General

We ignored the food terms that do not express any food class, such as dinner. Although some of these dishes contain certain ingredients most of the time, that does not mean they would not contain other ingredients. That is why, we can never be sure of them and they are not able to determine a certain food category. The full list is below:

Non-class food terms	alms, banquet, breakfast, broth, conserve, dainty, dinner, dumpling, feast, food, lunch, meal, morsel, preserve, provision, salad, soup, victual

Table 5. Ignore list for non-class food terms

3.1.1.2 Pet Names Similar to Food Items

We also found and ignored the most common food-resembling names for domestic animals. For example: Here, Ginger is coming back. It is a name for a cat with ginger color fur. (Note: these food terms will be ignored only when they occur in a capitalized form in the middle or end of a sentence after non *eat verbs*.)

Pet names resembling food terms	Apple, Bacon, Biscuit, Cherry, Cookie, Ginger, Honey, Kale, Kiwi, Mango, Olive, Peaches, Peanut, Tuna

Table 6. Ignore list for pet names resembling food terms

[51] Nguyen, Quynh C., et al. "Leveraging geotagged Twitter data to examine neighborhood happiness, diet, and physical activity." Applied Geography 73 (2016): 77-88.

3.1.2 Food Vocabulary Format

The list of terms for each food group and the verb "eat" synonyms were stored as a comma separated words in a text (.TXT) file.

3.2 *Folktale Corpus*

A corpus is a big collection of documents, in our case, folktales. Our corpus consists of 804 folktales.

3.2.1 Corpus Genre

We include all types of folktales such as animal tales, tales of magic and religious tales except the jokes and anecdotes (ATU type index 1200-1999)[52]. Our goal was to select folk stories resembling the bedtime stories or fairy tales the most.

3.2.2 Corpus Source

We compared two free online folktale corpora: Project Gutenberg and the collection on the website of the University of Pittsburgh[53]. The University of Pittsburgh collection was not arranged according to nationality or timing but according to ATU type. As a result, the entire data are inconsistent in terms of the number and choice of tales per nationality; certain nationalities we wanted such as Hawaiian and Australian were missing and other unselected nationalities were present. That makes it extremely difficult for us to create a corpus with folktales organized into different nationalities. Also, the collection contained written tales such as the ones by Hans Andersen. Another confusion arises from the type of collected items; they were sometimes links to tales and sometimes to external sources. We were also concerned about the public domain permissions as they were not explicitly shown. With the Gutenberg Project, all these concerns were relieved as long as we find the books in the nationalities we planned for.

The Project Gutenberg was founded in 1971 by Michael Hart with the aim of digitizing books for the public to use for free and it has now more than 50,000 electronic books, most of which are public domain, in widely used formats online in its collection as of now[54].

We decided to take opportunity of these books for our project and selected supposedly contrasting folktales for our computational analysis.

3.2.3 Collection Selection

For our project purpose, we needed to select folktales from contrasting cultures and geographical locations so as to train our classification systems with various types of food related texts. After numerous discussions and meetings, we set to collect stories from Australian, Dutch, German, Hawaiian, Indian, Japanese, Jewish, Portuguese, Russian and Spanish cultures for the time being (See Appendix C: List of Folktale Books for the Project

[52] http://mftd.org/index.php?action=atu
[53] http://www.pitt.edu/~dash/folktexts.html
[54] https://en.wikipedia.org/wiki/Project_Gutenberg

Corpus). Unfortunately, we could not find folktales of other cultures such as Persian we were interested in from the Project Gutenberg collections.

3.2.4 Collection Language and Timing

The language of the collection for our purpose was chosen to be English. We collected folktales from different cultures and locations in the common denominator language English and all folktales also had to come from around late 1800s and early 1900s also for normalization purpose. All 804 tales met these collection criteria.

3.2.5 Collection format

All tales were downloaded, separated and saved in the text (.TXT) format.

3.2.6 Data Annotation

We collected and annotated 804 tales from the Project Gutenberg because we did not have pre-labeled tales according to different dietary categories. This process took one person more than two months of time. The annotation process included thoroughly reading each and every sentence in every tale and filing the corresponding class labels and related food terms along with food related sentences into an excel file.

While annotating, we learned that certain food items perform not only food but also magical or impersonated roles. A fish can be eaten as a food in some tales, but it can be described simply as a companion or a magical creature in others. This phenomenon in fairy tales hinders us from categorizing the tales with those terms straight away; in addition we would consider ambiguous terms as food only when they occur as a direct object of *eat terms*. Based on this idea, we also needed to promote two additional notions in terms of referring to *food items*: *unambiguous food terms* and *ambiguous food terms*. The *unambiguous food terms* consist of the food groups *fruit, vegetable, grain, legume, dairy* and *unambiguous flesh*. The *ambiguous food terms* consist of the food groups *egg, honey, seed* and *ambiguous flesh* terms.

As it can be recalled, there are five different dietary classes. Omnivorous is the first and foremost category to check because our classifying rules run down hierarchically from the least restrictive food terms to the most restrictive terms. Once there are *unambiguous flesh* terms or *devourer* terms or the *ambiguous* terms coming after *eat verbs*, the tale is classified as *omnivorous*. Only the real monstrous flesh eaters are in the *devourer* list (See Table 4), the tales with the less monstrous beasts are considered omnivorous only when they are involved in consuming flesh; because in some tales, there appears a vegetarian wolf (in a modern retell of *the Little Red Riding Hood*[55]) or a crow (*A Crow and his three friends* from *Hindu Tales from Sanskrit*), so it is not a matter of what type of creature (unless the most monstrous carnivores), be it a tiger or a cave or an animal of the same species, is consuming foods, what matters is the food being eaten.

[55] https://www.goodreads.com/book/show/6431044-the-true-story-of-little-red-riding-hood

For the vegetarian class, not one occasion of dairy products was found being animated in any fairy tale in the corpus so far. This means all dairy products can be directly considered as *unambiguous food terms*. Therefore, we classify any fairy tale as *vegetarian* if there is a presence of any dairy product. In addition, eggs and honey are considered vegetarian as we mentioned previously. However, eggs have many roles in tales from being a life-storing safe or animal's young to becoming a precious golden piece, so we will categorize eggs only if it is the object of any e*at terms*. Same principle applies for honey as they are ambiguous when used as an endearment term.

For the class vegan, the presence of grains, legumes, or vegetables makes the tale automatically vegan. In folktales, these are rarely animated and mainly considered as food by default. There were only two occasions in the entire corpus: an animated bean in *Why the Beans Have Black Spots* and gingerbread in *Why the Pigs Root in the Mud* (both Dutch folktales). Even if they are eaten, they still would not hurt the feelings of vegans as much as the flesh products do. Therefore we can classify the tale as vegan safely no matter they are animated or not. Similar to the vegan classification, fruits are usually unambiguously foods and we have not found any animated fruits so far. And eating fruits would not hurt any feeling even if animated. However, the term *seed* can also have different roles in tales, so this is also an ambiguous term and should be considered as food after *eat verbs.*

With these notes from the annotation process, we created and followed the rules in Figure 7. For the rule-based classifier, the classes are really defined by the presence of those terms belonging to the class covering the widest possible scope (See Table 7). For example: if the tale contains the terms *milk*, *apple* and *cabbage*, then the class would be *vegetarian* no matter it contain *vegan* or *fruitarian* terms because *milk* belongs to the widest-scope class *vegetarian*. If we add *meat* to this set, then the tale turns to be *omnivorous*. In *vegan* tales, *omnivorous* and *vegetarian* terms are absent. In *fruitarian* tales, all three higher class terms are absent. When there is no food term present in the tale, then the tale is a neutral one.

		Term scope			
		omnivorous	vegetarian	vegan	fruitarian
Classes	Omnivorous	+	+		
	Vegetarian	-	+	+	
	Vegan	-	-	+	+
	Fruitarian	-	-	-	+
	Neutral	-	-	-	-

Table 7. Class decision structure

Figure 7. Hierarchical rules for labeling folktales.

After the annotating process, there were 33 fruitarian (f), 163 vegan (v), 44 vegetarian (vg), 262 neutral (n) and 302 omnivorous (o) tales.

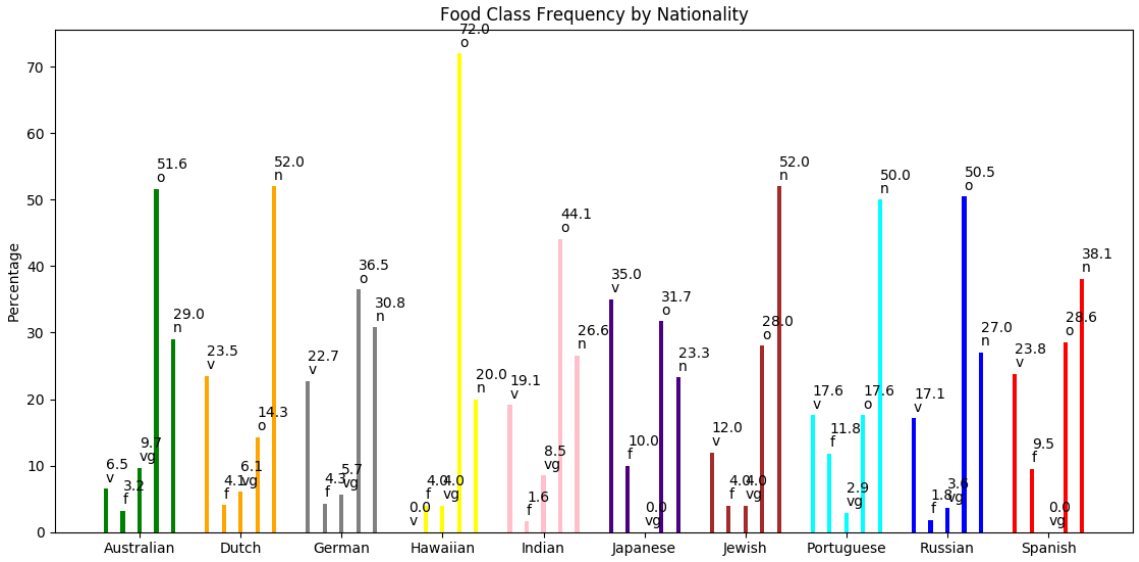

Figure 8. Occurences of each class per nationality

In general, there were certain expectations about certain nationalities in terms of food cultures. Some of these met: for example, the Japanese, the Portuguese and the Spanish are historically known as not consuming heavy dairy products and it was reflected correctly in our analysis with 0% occurrences in their vegetarian tales while the dairy consumption were high among the Australian (aboriginal) and Indian natives, 9.7% and 8.5% respectively. The presence of vegan tales was the most prevalent among the Japanese (38%), Dutch (23.5%), Spanish (23.5%) and German (22.7%) and these nationalities are already well-known heavy grain and vegetable consumers. As you can see, the folktales from the European and Asian cultures, including Russian, in our corpus are consistently weighty in vegan foods, possibly, because the diet simply consisted of basic food due to poverty and scarcity in reality not due to ideology. In other words, people might not have chosen to live a vegan lifestyle as they wish like in today's world but simply they had no other food choice. The terms *vegan* and *vegetarian* are still new in many cultures except those in countries such as India. The stories from the temperate Southern Europe contained the highest amount of pure fruits: Portuguese (11.8%) and Spanish (9.5%) respectively and this also reflects our prior predictions very well. On the flip side, there were also opposing results in our analysis. Contrary to prior expectation, the tropical groups of Hawaiian, Australian and Indian groups had a large proportion of omnivorous tales, 72%, 51.6% and 44.1% each, due to frequent mentions of man-eating monstrous or cannibalistic terms, specifically in the former two. If these nationalities had contained high fruit tales, then our dataset would have been more balanced. The neutral tales, which contain no food related terms, maintained high frequency throughout the corpus, ranging from 20% in Hawaiian up to 53.1% in Dutch. Interestingly, the Dutch and Jewish stories had similar patterns although it could be of pure conincidence. None of the nationalities had balanced or close to balanced distributions of the five classes.

These varying results could be due to a number of possible reasons. Perhaps, there could be a gap between the traditional diets and modern human concepts about diets. Or certain folktales could simply be mirroring the rare and expensive foods as it is common among folktales to reflect dreamy and wishful contents overcoming miserable, poor and hungry states. Or these could be mere coincidences as folktales were not made to focus on foods.

4 Methodology

4.1 *Tools and Libraries*

We listed all the tools and libraries for our project in the following table. And we included a brief introduction to the core tools below the table.

Programming Language	Other Development Tools	Deployment Tools
Python 3.5	Scikit Learn, NLTK, Pickle, Chunking, Stanford CoreNLP, Stanford POS Tagger, CoreNLP Pywrap, Openpyxl, Java jdk 1.8	PythonAnywhere, Flask

Table 8. Main tools and libraries used in the project

4.1.1 Python

Python[56] is one of the most competitive programming languages these days and its intuitiveness of both human and computational languages made it popular in machine learning and natural language processing. Moreover, Python supports many open source libraries that we use extensively in computational linguistics.

4.1.2 Scikit Learn

Scikit Learn[57] is a free machine learning library for Python[58] and accommodates most of the popular machine learning algorithms from which we built our classification models. Its superior integration with Python allows us to use it along with other popular libraries for language analyzing. Also, the predictive models created in this module can be deployed online through many Python frameworks.

4.1.3 NLTK

Natural Language Toolkit[59] (NLTK) is a well-known Python platform dedicated for human language analysis. The module provides us with text processing libraries such as tokenizers, lemmatizers and parsers.

[56] https://www.python.org/. Version 3.5
[57] http://scikit-learn.org/
[58] https://en.wikipedia.org/wiki/Scikit-learn
[59] http://www.nltk.org/

4.1.4 Openpyxl

Openpyxl[60] is a Python library to read from and write to Excel files. We use it to read the "gold" (true) folktale classification labels from our annotation file.

4.1.5 Pickle

The Python Pickle[61] module is used to serialize and de-serialize Python object structures. This possibility of portability allows us to store data from runtime structures and open the stored runtime data in another Python script with ease.

4.1.6 Chunking

Chunking[62] is a technique from the NLTK module to extract information from the given text following certain patterns. It is used to extract phrasal verbs in our project.

4.1.7 Stanford CoreNLP

Standford CoreNLP[63] contains a range of grammatical analysis tools written in a versatile programming language JAVA[64]. For our project, we used its Coreference Resolution system[65] to solve the anaphora problem in our rule-based classification because we found this is the only coreference resolution tool that can be applied in our Python system without much complication of being outdated, using different version of Python or other compatibility issues, throwing multiple errors and not working at all and so on. Plus, this is the winning system at the CoNLL-2011 shared task coreference evaluation and has since added the advantages of statistical and neural coreference solutions for better performance (at the cost of higher memory requirement of at least 4GB).

4.1.8 CoreNLP Pywrap

CoreNLP Pywrap[66] is a Python language wrapper for Stanford CoreNLP so that we can use the anaphora resolution module in our Python project because Stanford CoreNLP is a toolkit that originally runs in a JAVA environment. This wrapper had the most complete and working interpretation of the original Stanford CoreNLP among the other wrappers we tried such as Stanford CoreNLP Python[67], PyCoreNLP[68], StandfordCorNLP[69] and Stanford CoreNLP[70].

---

[60] https://openpyxl.readthedocs.io/en/default/
[61] https://docs.python.org/3.2/library/pickle.html
[62] http://www.nltk.org/book/ch07.html
[63] https://stanfordnlp.github.io/CoreNLP/
[64] https://www.java.com/en/
[65] https://nlp.stanford.edu/software/dcoref.shtml
[66] https://github.com/hhsecond/corenlp_pywrap
[67] https://github.com/dasmith/stanford-corenlp-python
[68] https://github.com/smilli/py-corenlp
[69] https://github.com/Wordseer/stanford-corenlp-python
[70] https://github.com/Lynten/stanford-corenlp

4.1.9 Stanford POS Tagger

Stanford POS Tagger[71] is a part of speech (POS. See Appendix B: Part of Speech Tags) tagging module from the Stanford team and is known to be most effective in part of speech tagging. This was confirmed on our test run on the original *Rapunzel* text and out of 528 unique pos tagged terms (total 1693 pos-tagged terms in the text), there were 103 unique instances that were tagged differently in Stanford POS Tagger and NLTK POS Tagger. From the 103 unique sets, the Stanford system tagged 64 (61.54%) correctly and the NLTK one tagged merely 25 (24.27%) correctly and both tagged 14 incorrectly. Therefore, we decided to use it instead of the NLTK tagger as it showed higher performance than the latter from the above example. (Note: we later developed our coding not to rely heavily on automatic pos-taggers and lemmatizers, so it would not matter much which tagger was used)

4.1.10 Flask

Flask[72] is a Python based web microframework and its simplicity and lightweightedness convinced us to employ it to deploy our offline machine classifier to an online environment.

4.1.11 PythonAnywhere

PythonAnywhere[73] is a cloud-based Python development environment. It is fully supported and the most, and probably the only, Python friendly online hosting environment up to date. We host the machine classification part of our project on its free web hosting plan.

4.2 *Computational Classifications*

Both rule-based and machine learning classifications are implemented by the means of computational devices. Hence, we regard both of them as computational classification methods. The former one follows human-written rules and the latter one learns by itself.

We all know that statistical machine learning approach has its own shortcomings and its performance depends largely on the size of the corpus or quality of the data. Alternatively, the traditional rule-based approach does not need any training as long as it has good rules to follow. In certain situations where the category of the text depends on only one or so words, the rule-based method shines in. Also due to the hierarchical nature of the dietary classification, we developed the rule-based system along with the hybrid machine classification system (See Table 7. Class decision structure) because conditional checking is the basis of rule-based systems.

Machine classification is an algorithm that learns how to divide the mixed collection into two or more different classes based on the classification of the training data and then evaluates on the testing data. This facilitates human work tremendously on the condition that we have good data.

---

[71] https://nlp.stanford.edu/software/tagger.shtml
[72] http://flask.pocoo.org/
[73] https://www.pythonanywhere.com/

As we develop both types of classifiers, we were wondering which would outperform the other.

4.2.1 Text Preprocessing

Our project was designed for English documents from the beginning and all our corpus documents are converted to ASCII as we did not want to include any character from the Latin-1 supplement, which contains specific characters mostly from Western European languages. Due to the fact that our tales come from different translators and writing cultures, we found some of the texts contained certain characters such as curly quotes and accented letters that are processed differently on different platforms. For example: Several tales contained curly single and double quotes and Python on Windows had no problem handling it properly; however, the same version of Python on Unix based systems had a problem with it. And also, some of our tools were also not able to process those characters as they have different length than the Basic Latin characters. As a result, this difference had an ill effect on the coding compatibility across different platforms; therefore we had to replace them with the ASCII equivalents. (See Appendix A: Latin-1 Supplement Character Set with Replacement Equivalents)

Spaces: Replaced all different types of space such as tabs and operating system specific line breaks and a string of multiple spaces with a single whitespace.

Bracketed notes: Removed the redundant explanation texts in between square and oval brackets, that are not the original parts of the tales themselves.

Abbreviations: Replaced common abbreviations with their corresponding words: *I'm > I am*.

Archaic English terms: Replaced the archaic English terms: pronouns, such as *thee* and *thou*, modals such as *mayst* and *canst*, adverbs such as *hither* and *whither*, and certain verbs such as *eatest* and *cookest*, with their modern counterparts as there is a high chance that they might not be recognized by the modern part of speech taggers.

End of sentence: Normalized sentence endings with a *dot*.

Specific punctuations: Removed the specific punctuations, that do not partake in the standard sentence structure, such as # and *.

Pet names: There are certain food terms used to name domestic animals. For instance, Ginger, Cookie and Honey (See Table 6). We needed to remove them (following certain rules such as when they are capitalized and in the middle of a sentence) before the search rules applied.

Possessive *S*: All nouns with the possessive *S* immediately after them are removed with the possessive *S* as they act only as modifiers and have no role with finding the food related nouns.

Lemmatization: Lemmatizing usually involves tokenizing text into separate terms called tokens, and POS tagging each token, and then lemmatizing the inflected variants of certain

word groups into their original base form. We used customized lemmatization for both classifications. Lemmatization brings us the uninflected dictionary forms of words. For example: *be* is the lemma for *was, were, am, is, and are* and *eat* is the lemma for *ate and eaten*. Lemmatization makes the verbs lose tenses and nouns lose countability so that we can search the lemmatized text using the base forms of words instead of going through hundreds more *if…, then…* checkings. So, this is also a way of reducing and facilitation our search function.

After implementing part of speech tagging, we then can lemmatize the given text based on the tagged information. As we have now already realized, we need *unambiguous food terms* and *ambiguous food terms* following *eat terms* to classify a certain tale. Therefore, we are interested in looking for the lemmas of the *food* related nouns such as *orange* and *coconut* and *eat* related verbs such as *swallow* and *devour*. However, if these terms are inflected in various tenses and counts, it would be really hard for us to search and find them as mentioned above.

With our lemmatization function, we lemmatize all verbs and nouns and remove custom stop words (See Table 9) and ignore all kinds of modifiers. For example, a sentence *He ate all those apples* would turn into *He eat apple* after lemmatizing by fixing the verbal tense of the *eat term*, the noun plurality and the redundant modifiers before the actual food term we are after.

Once we have lemmatized tales, we can search for the terms in Table 2 and Table 3 and select the sentences we need. After this selection, for example, the *Rapunzel* story from Brothers Grimm stories becomes as following:

rampion. She make herself salad of it , eat salad of it (the anaphora resolution tool replaced eat it with eat salad of it).

Stop words: Stop words are terms that occur too frequently in a text, such as *a*, *an* and *the*. These words do not carry any meaning for most linguistic analyses. We used a custom list based on the English stop words from the NLTK library for both machine and rule-based classifications. From the 127 NLTK stop words, we removed all the pronouns as they are necessary for our classification and added folktale specific common words such as *once, upon* and *time*. (See Table 9)

The above mentioned clean-ups first performed to make it easier to deal with pure nouns and verbs later on.

	Stop words
NTLK	a, about, above, after, again, against, all, an, and, any, as, at, because, before, below, between, both, but, by, did, do, does, doing, down, during, each, few, for, from, further, had, has, have, having, here, how, if, in, into, just, more, most, no, nor, not, now, of, off, once, only, or, other, out, over, own, same, so, some, such, than, that, the, then, there, these, this, those, through, too, under, until, up, very, what, when, where, which, while, who, whom, why, with
Folktale	ago, away, bad, castle, country, curse, ever, evil, far, good, happily, hut,

specific	kingdom, live, lived, living, long, magic, palace, place, spell, state, time, tomorrow, upon, yesterday

Table 9. Custom stop words that need to be removed from the vocabulary

4.2.2 Further Processing

We tried to process the tale texts further by replacing the anaphora with their antecedents and removing redundant prepositional phrases. However, both functions showed poor performance and we decided not to implement either in the end. The further text processing is in detail below.

4.2.2.1 Anaphora resolution

Many instances of the anaphora resolution problem are found throughout the corpus tales in the form of:

eat verbs (Table 1) + personal object pronouns[74]

Out of the seven different object pronoun options, the *it* and *them* are the ones that need to be replaced with their antecedents. The other five: me, you, us, him, her, almost always were indicating a flesh eating activity if they follow one of the *eat verbs*, so we could assign them an omnivore tag. However, when there is a sentence *I will eat it.* or *he devoured them.*, we still have no clue as to whether the object is food or not.

We applied the Stanford CoreNLP coreference resolution module on our *eat verb + it/ them* sentences. As we are interested in only the eat verb + it/ them sentences, we first extract these sentences after removing the phrasal verbs and lemmatizing the tales. Out of all 147 anaphoric instances, containing *the eat* or *food terms*, in 102 distinct tales from the entire corpus, the furthest antecedent was five sentence away from the anaphora. We initially set the distance boundary to 2-5 sentence away, however, we realized that the tool was restricted and utilized any antecedent found within that distance. As a consequence, we ran the anaphora resolution tool without setting any specific distance boundary supposing the tool would find the antecedent successfully.

Based on our analysis in the results, we assume that even the best coreference tool is not able to decipher the anaphora problem at all if the antecedent is too metaphoric and not explicitly mentioned as in the case of a Russian tale *Prince Ivan, the Witch baby and the little sister of the Sun*. The antecedent in the tale (as *she will eat up your father, and eat up your mother, and eat up you too*) is found in the sentence 13, which is 90 sentences further ahead of the anaphora (*eaten them*) in sentence 103. Or the antecedent could have never been mentioned or the sentence structure could be too complex for the tool among other impossibilities. For

[74] Personal object pronouns: me, you, him, her, it, us, them

instances such as these, the tool would throw *timed out* exception after its exhaustive search. Such cases made up 25.85% of the total 147 anaphoric instances.

For other instances, even when the antecedents are found, they were not deciphered at all (13.61%) or improperly detected (46.94%). Only 13.61% was correctly matched. The anaphora resolution system particularly struggled with finding the antecedents for *them* instances (only 4.17% correct) than *it* instances (18.18% correct).

Overall, the anaphora resolution performance improvement was nearly non-existent, in fact, its implementation with a rule-based test run on the entire corpus using the specific food related features for both classifiers, the metrics improved almost negligibly (both accuracy and f measure by mere 0.001). Therefore, we decided we would rather not to use it regarding both performance and complexity.

4.2.2.2 Chunking for Prepositional Phrase Removal

A soft or partial parsing technique called *Chunking*[75] allows us to selectively extract phrases based on the custom grammar rules we write. This method uses Regular Expression and Grammar tags to extract specific phrases from a text.

Since prepositional phrases usually add extra information to sentences, we assumed it might improve the corpus wide search and classification functions if we erase those redundant bits.

The chunk grammar we create to filter prepositional phrases is:

{<TO|IN|RP><DT|CD|PDT|PRP\$|JJ.*>*<NN.*>+}

<TO|IN|RP>: find all possible prepositions followed by

<DT|CD|PDT|PRP\$|JJ.*>*: optional determiners or numbers or predeterminers or possessive pronouns or adjectives followed by

<NN.*>+ one or more nouns.

Although we aim to remove all the prepositional phrases in the whole tale so that the sentences in the tale can be reduced effectively for improved further search and ambiguity efforts, if the phrase contains an unambiguous food term such as fruit, potato, milk, or bacon as a noun then we need to keep those terms as they could determine the category of the tale. For example, a prepositional phrase *into the magical fruit forest* from a sentence *She went into the magical fruit forest* is as following after POS tagging: *into(IN) the(DT) magical(JJ) fruit(NN) forest(NN)*. Although, there is an unambiguous food term *fruit* is in this phrase, this is an adjective that is being used to just modify the head word of the phrase. The head word of this phrase, *forest*, is not in the list of our food items, so we can safely remove the whole phrase from the sentence. In another example *She took the ring from the fruit*, we would still remove the phrase *from the fruit* from the sentence but the unambiguous food term *fruit* is a

[75] Bird, Steven, Ewan Klein, and Edward Loper. Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.", 2009.

noun in this case, so we should store the term and put it at the end of the tale and then remove the phrase. A toy folktale excerpt is shown below:

Once upon a time there was a princess who loved fruits. One day, she went into the magical fruit forest nearby. And she came to a pond and saw a glowing, transparent apple on the ground. When she took it off the ground, it had a beautiful ring in it. She took the ring from the fruit. It was a wishing ring!

After prepositional phrase removal using the chunking method, the tale is reduced to:

Once there was a princess who loved fruits. One day, she went nearby. And she came and saw a glowing, transparent apple. When she took it, it had a beautiful ring. She took the ring. It was a wishing ring! fruit.

The last word *fruit* is the term we stored from the phrase *from the fruit* after removing it.

The removal of the prepositional phrases did not meet our expectation of improving the scores, in fact, it was the opposite of what we expected. The accuracy scores decreased by 0.006. We assume it is because the chunking process itself relies heavily on the part of speech tags; hence, it cannot perform better than the tagger. From our random trial of tagging the tale *Rapunzel* using the Stanford POS Tagger in section 4.1.11, it is clear that even the best tagger might predict only the 61.54% of the specified terms correctly. It could be the main culprit.

4.2.3 Rule-based Approach

While annotating the tales, we filed and analyzed all the sentences and phrases related to foods and eating actions and finally made generalization rules over in what pattern they usually occur (See Figure 7).

In our rule-based classification, we basically used the methods of reducing and filtering mostly to reach the final conclusion. To make the classification decision for the given tale, we need to find the occurrences of the unambiguous food terms or the eat sentences containing the ambiguous food term.

4.2.3.1 Classification

After reducing the text and getting rid of unnecessary load of information, and then lemmatizing nouns and verbs using the above methods, we are finally able to check if the tale contains the terms we are after.

With unambiguous food terms, we can assign a class to the tale straight away depending on what food group the term belongs to. However, we also need to check the ambiguous terms as direct object of *eat terms*. To do so, we extract the sentences containing *eat terms*. Then we check the next word after it. If the next word is any of the flesh terms or upper case terms for a possible human name, reflexive pronouns or object pronouns except it and them, which should have been resolved and replaced with the anaphora resolution tool, then that sentence is considered an omnivorous sentence. If the tale contains any omnivorous sentence or any unambiguous flesh term or monstrous creature such as *cannibal*, *man-eater*, *ogre* or

rakshas[76], then we classify that tale as an omnivorous tale. If none of these is found, then we can step down to check if the tale contains any of the unambiguous vegetarian terms in addition to the two ambiguous vegetarian foods, *egg*s and *honey*, as a direct object of *eat terms*. If so, that tale is classified as a vegetarian tale. There is no ambiguity in vegan tales, so we can simply check if the tale contains any of the vegan terms and classify accordingly. For fruitarian tales, if we did not find any of the food terms belonging to the previous three classes, then we search for unambiguous fruits and nuts or the ambiguous seeds as a direct object of eat terms, then classify accordingly as above. If the tale does not contain any food related term or sentence, then that tale is assigned a neutral class.

4.2.4 Hybrid Machine Learning Approach

In machine learning, there are two kinds of tasks: supervised and unsupervised. Our problem is a supervised task because we have a pre-labeled corpus while unsupervised tasks learn from unlabelled data. And it is a classification problem for we are going to guess a class (e.g. *vegan*), not a number (e.g. 1) for fairy tales. We are going to predict from more than two different classes, therefore it is a multiclass classification whereas binary classification solves two category problems. In multiclass problems, each sample is assigned to one (and only one) label.

4.2.4.1 Imbalanced dataset

Our now supervised multiclass classification task has a balancing problem. When the class distribution is uneven, there is usually a problem of unbalanced data. Our corpus of 804 folktales has unbalanced distribution (the class fruitarian has 26 instances, vegetarian 42 vs. omnivorous 300, neutral 275 and vegan 161 respectively). There are various ways to deal with this problem out of which we decided to implement the cost effective *sampling method* along with certain other trivial strategies such as setting the classification algorithm specific parameter *class_weight* to *balanced,* choosing the evalutation metric as *f measure* with the *averaging* parameter to multiclass friendly values *weighted* or *micro* and implementing *stratified allocation* of instances for cross validated metrics.

4.2.4.2 Evaluation methods

For multiclass classification problems, we can use accuracy as our evaluation metric if all the classes are balanced out. However, when there is overtaking of one or more classes and our classifier simply favors those majority classes, then we get really high accuracy without even attempting to classify the other minority classes. For example, a dummy baseline classifier that assigns *omnivorous* to all predicted labels when the 80% of the entire corpus is labeled as *omnivorous*, then we achieve 80% accuracy. This is known as the Accuracy Paradox[77] and therefore, we were not able to use accuracy as our evaluation measure for our imbalanced classification problem. But we used accuracy on our dataset once balanced it out.

[76] Rakshas or Rakhas are man-eating creatures that look like normal humans in Indian folktales
[77] https://en.wikipedia.org/wiki/Accuracy_paradox

We used the dummy classifier as our baseline for the imbalanced dataset; but for the balanced dataset, we used the random baseline model as baseline. The random baseline model assumes that the class distribution is equal in the corpus.[78] And if the dataset had 5 classes, for example, the Random baseline accuracy would be 1/5 = 20%. This is why it is recommended for balanced dataset.

With the imbalanced dataset, we relied more on f-measure[79, 80] to make decisions. The f-measure or f1 score is the harmonic combination of precision, the number of corrects out of the predicted corrects, and recall, the number of corrects out of the real corrects. For instance, suppose there were 15 apples and 10 oranges, and the classifier predicted 14 fruits as apples and 11 as oranges; however, out of the predicted 14 apples, 12 were correct, then the precision for apples is 12/14 and the recall for apples is 12/15. The 12 is what is called True Positive (TP), the 2 oranges misclassified as apples are False Positives (FP). The 3 apples misclassified as oranges are False Negatives (FN). Then the formulae[81] are as following:

$$P = \frac{TP}{TP+FP} = 0.86 \qquad R = \frac{TP}{TP+FN} = 0.8 \qquad F1 = 2\frac{PR}{P+R} = 0.83$$

Accuracy would be percentage of corrects out of total:

$$A = \frac{TP+TN}{TP+FP+TN+FN} = \frac{12+8}{12+2+8+3} = \frac{12+8}{25} = 0.8$$

For more than two classes, the precision and the recall need to be averaged by one of the four options in SK Learn, namely, *macro*, *micro*, *weighted* and *sample*. The last one is reserved for multi-label problems that assign more than one class to a data point. The other three averaging methods are explained below.

Macro

Macro weighted scores are weighted by the number of correct instances for each class. This averaging takes the sum of all precisions or recalls for each class and then divide by the number of classes. The f-measure then simply is a harmonic mean of these two. In another way, it's the arithmetic mean of the all f-measures for each class. This calculation method gives equal weight to each class without caring for the size of the classes, thus, it can be suitable for balanced problems. For highly skewed data, the macro averaged f-measure tends to be lower due to the fact that the minor classes are harder to classify and are overemphasized by this method. Therefore, we can use this averaging only after we have balanced out our dataset.

---

[78] Ahn, W. Y., Busemeyer, J. R., Wagenmakers, E. J., & Stout, J. C. (2008). Comparison of decision learning models using the generalization criterion method. Cognitive Science, 32(8), 1376-1402.

[79] https://en.wikipedia.org/wiki/F1_score

[80] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[81] https://sebastianraschka.com/faq/docs/multiclass-metric.html

$$P_{macro} = \frac{\sum_i^k P_i}{K} \qquad R_{macro} = \frac{\sum_i^k R_i}{K} \qquad F1_{macro} = \frac{\sum_i^k F1_i}{K} \text{ or } F1 = 2\frac{PR}{P+R}$$

Weighted

Weighted averaging scores for each class are weighted by the size of the classes. This averaging accounts for class imbalance problem by computing the average of binary metrics in which each class's score is weighted by its presence in the true data sample.[82]

$$P_{weighted} = \frac{\sum_i^k N_i P_i}{N} \quad R_{weighted} = \frac{\sum_i^k N_i R_i}{N} \quad F1_{weighted} = \frac{\sum_i^k N_i F1_i}{N} \text{ or } F1 = 2\frac{PR}{P+R}$$

This averaging method reduces the misclassification effects caused by the infrequent classes.

Micro

Micro averaging calculates the metrics globally and weighted by class distribution. This computes precision and recall by summing up the individual TPs, FPs and FNs for each class. Micro-averaging may be preferred in multilabel settings, including multiclass classification where a majority class is to be ignored.[82][83].

$$P_{micro} = \frac{\sum_i^k TP_i}{\sum_i^k TP_i + FP_i} \quad R_{micro} = \frac{\sum_i^k TP_i}{\sum_i^k TP_i + FN_i} \quad F1 = 2\frac{PR}{P+R}$$

Note that for single-label multiclass classification, the total number of false positive decisions is the same as the total number of false negative decisions and hence, the micro-averaged F1 is identical to the other commonly used measures: accuracy, micro-averaged precision and micro-averaged recall.[84]

As we can see, it is suggested that the meaningful averaging methods of the f-measure for imbalanced problem would be ***weighted***. The ***macro*** is usually reserved for balanced dataset and the *micro* is more meaningful when the majority class is out of context. We always kept the majority class.

To statistically compare the performance of different models, we used McNemar's Chi Square test when comparing two dependent classifiers, Cochran's Q when comparing more than two dependent classifiers and Z test with two proportions when comparing two independent classifiers.

---

[82] http://scikit-learn.org/stable/modules/model_evaluation.html

[83] Sokolova, Marina, and Guy Lapalme. "A systematic analysis of performance measures for classification tasks." *Information Processing & Management* 45.4 (2009): 427-437.

[84] Van Asch, Vincent. "Macro-and micro-averaged evaluation measures [[basic draft]]." (2013).

4.2.4.3 Text Representation in Machine Readable Format

We need to convert ordinary words into numbers, specifically, numerical feature vectors of fixed size, so that the computer classification algorithm can understand our data as they are not able to deal with raw text data of varied size. [85]

We first implemented Count Vectorizer which simply counts the occurrences of each term per document (or folktale) as its name suggests and creates a sparse feature vectors as shown in the following table. This vector matrix cares only the counts, not the order, of the terms, therefore it is also known as *Bag of words* among linguists.

Document\Term	once	Upon	The	….	chocolate
Rapunzel	7	0	89	…	0
Cinderella	4	0	230	…	0
…	…	…		…	…
The Chocolate house	2	1	59	…	4

Table 10. Counter Vectorizer feature matrix

When the number of documents increases, certain uninteresting words such as *the* becomes frequently present with their meaningless weights. The Term Frequency Inverse Document Frequency (tf-idf) is one very effective method to solve this problem. The tf-idf calculation gives more weights to the more important and meaningful terms such as *chocolate* and less weights to the frequent terms such as *once*. The tf-idf weight is the product of two parts: first, the term frequency (tf) is the count of term *t* in document *d* normalized by the count of total words in the document, second, the inverse document frequency (idf) is the log of the count of documents *D* in the corpus over the count of documents where the term *t* appears in.

tf-idf (t, d, D) = tf (t,d) * idf (t, D)[86]

Before this conversion happens, we need to decide on figure how many grams of words go together. In the *bag of words* approach, *unigram* is when we have individual orderless words in the bag, *bigram* is two sequential words are individualized and *tri* or *more grams* are such number of sequential words are individualized. In an extreme example of higher order grams, if you let fifteen such sequential words to stick together, that makes a size of an average English sentence[87] and you are unlikely to encounter the same structure in the entire corpus. That is why, uni-tri grams are often used in practice and it is suggested to go for lower order grams, especially when you have a small dataset. If you have high variance data such as in text documents, then the issue of overfitting unanimously follows, therefore it was wise for us to choose less than tri grams although we tested up to that point.

4.2.4.4 Feature extraction

We tried a few different feature extraction variants. First we tested the original text with the simple Naïve Bayes after the initial clean-ups (average number of words per tale = 988.4). However, it was too noisy and the machine was not learning well (See Table 14). Further, we

[85] http://scikit-learn.org/stable/modules/feature_extraction.html
[86] http://scikit-learn.org/stable/modules/feature_extraction.html
[87] Cutts, Martin. *Oxford guide to plain English*. OUP Oxford, 2013.

let only nouns and verbs stay (average number of words per tale = 590.0), but it was still not much improvements (See Table 15). And last we selected only the related nouns and verbs (average number of words per tale = 4.7) and it started looking better (See Table 16). Therefore, we chose to proceed with the related features only and an example of the extraction procedure on a folktale *Rapunzel* is illustrated in Appendix D: Feature Extraction.

For machine classification, we used bag of words approach where the n-gram words appear independently. As we rely on the machine to learn by itself, we had to perform additional removal. We are now fully aware that we are after specific nouns (food terms) and verbs (eat terms). And many folktales contain no or a few of these terms. In such conditions, it is not very practical to expect the machine to learn from almost negligible amount of terms. Consequently, we tested with the entire text, only nouns and verbs, and last only the nouns and verbs we are interested in. The last option showed us the best performance.

4.2.4.5 Classifier Choice

There are many machine learning algorithms to choose from and we are interested in the ones that can handle multiclass classification for a small and imbalanced dataset of 804 labelled folktales with high dimensionality. The algorithms that can solve multiclass problems inherently are: Multinomial Naive Bayes (NB), Decision Trees (DT) and K Near Neighbors (KNN). However, most modern classifiers, especially the ones in the up-to-date Scikit Learn library, support multiclass classification by default; the Support Vector Machine (SVM) representative algorithms in Scikit Learn, the SVC-Supper Vector Classification, NuSVC-Nu Support Vector Classification and LinearSVC-Linear Support Vector Classification, are the classes that are capable of performing multi-class classification on a dataset by default[88].

Moreover, most text classification problems are linearly separable [89].(See Figure 10) That let us automatically remove nonlinear algorithms such as Neural networks (NN), DT, and KNN. It is also suggested not to use complex algorithms with many hyperparameters for small datasets, especially if they are imbalanced, for the complex classifiers tend to overfit.

And we are left with the linear options: Linear SVM and The Multinomial NB[90].

[88] http://scikit-learn.org/stable/modules/multiclass.html, Raschka, S. (2015). Python machine learning. Packt Publishing Ltd.

[89] Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." *Machine learning: ECML-98* (1998): 137-142.

[90] http://sebastianraschka.com/Articles/2014_naive_bayes_1.html

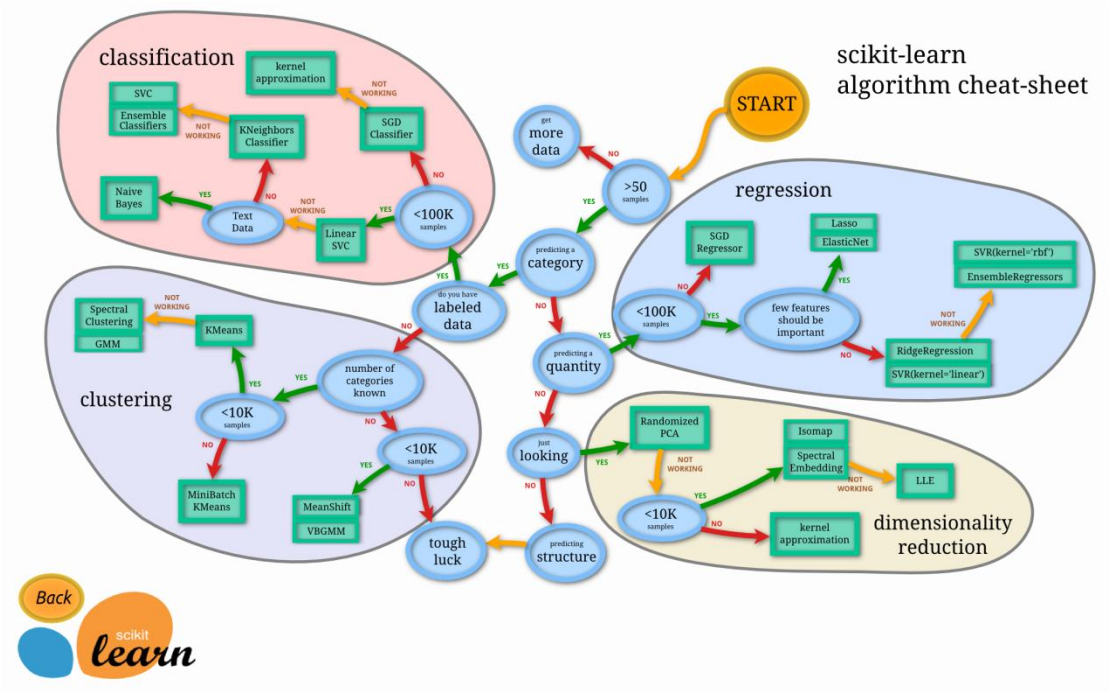

Figure 9. Classification algorithm map from SK Learn[91]

The SK Learn decision map in Figure 9 to help us make our mind to select the most suitable algorithm for our problem clearly suggests to choose the Linear SVM for classification problems with small data sets and if SVM fails then go for NB for text classifications leaving out the rest. In fact, the non-linear models, KNN, DT and NN, are designed to define the separating points known as decision boundaries accurately without finding good overall generalized patterns, especially for small datasets it happens very easily, because those algorithms naturally have complex interactions among the features.

Text classification problems mostly have a large amount of instances (documents) and features (words), which is also defined as high dimension. When there are immense amount of features to learn from, creating higher dimensions, the non-linear models overfit by simply memorizing and not predicting new data well. [92] [93] [94] This is known as the *Curse of Dimensionality*[95]. Although KNN could be suitable for a small dataset, it is not recommended for text classification in the image above. Because, as feature dimensionality increases as always in text classification, the distance to the nearest neighbor approaches the distance to the farthest neighbor. In other words, the contrast in distances to different data points

[91] http://scikit-learn.org/stable/tutorial/machine_learning_map/

[92] Spruyt, Vincent. "The Curse of Dimensionality in classification." N.p., 16 Apr. 2014. Web. 15 June 2017. http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/.

[93] Brownlee, Jason. "Overfitting and Underfitting With Machine Learning Algorithms". N.p., 21 Mar. 2016. Web. 10 July. http://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/

[94] Aggarwal, Charu C., and ChengXiang Zhai, eds. *Mining text data*. Springer Science & Business Media, 2012, pp. 200.

[95] Parker, Charles. CS534-Machine Learning: The Nearest Neighbor Algorithm[Slide #13]. Oregon State University. 2005. Retrieved from: https://web.engr.oregonstate.edu/~tgd/classes/534/slides/part3.pdf

becomes nonexistent and the algorithm struggle hard to make decisions. [96] Similarly, DT suffers from finding a good node to split due to the dimensional expansion and start overfitting, for small datasets, but the bigger problem with DT is the model instability issue. The model instability refers to a phenomenon where a slight change in training data leads to large changes in the resulting tree, hence, the resulting scores, and this is caused by more than five classes and high dimensional data, both of which we have in our task[97]. Furthermore, the instability of decision trees can be increased by using small size datasets[98]. When we ran a test trial, the instability of DT was observed producing different results for each run when there was no change in the dataset. It is known that DT and NN are unstable and Linear models and KNN are stable.[99] Although certain neural network algorithms could handle the *Curse of Dimensionality*, they need even more data points on top of the required datasets with thousands of samples to train. In short, when there is a possibility to solve the problem using a simpler method, it is suggested that there is no need to deal with a complex algorithm with complicated settings that requires bigger memory, computational efforts and other resources.

While these are already known directions passed down from the earlier research literatures, then experimenting with all possible learning models would be inefficient and resource challenging. Rather, we should *stand on their shoulders* and explore the most suitable among the suggested options.

For all the above numerous overlapping reasons, simpler and linear models, such as SVM and NB, are recommended for multiclass text classification with a small and imbalanced training dataset[94].

The SVM can be both linear and nonlinear, with *linear* and *rbf* kernel settings in the Scikit Learn.[100] It is well known that the SVM is considered as very effective for the high dimension feature problems[101]. Although, SVM is highly inefficient to train for a huge corpus, we have less than 10,000[102,103] instances (804 labeled folktales), therefore the primary SVM algorithm out of all possible options in Scikit learn[104] is the one we would prefer because our corpus is not large so it would not take too much time.

The Multinomial NB works well with a small corpus but since it is a very simple and naive classifier, it does not handle unbalanced dataset[105] well. Nevertheless, we experimented with both the Linear SVM and NB.

---

[96] Beyer, Kevin, et al. "When is "nearest neighbor" meaningful?." *International conference on database theory*. Springer, Berlin, Heidelberg, 1999.

[97] Baranauskas, José Augusto. "The number of classes as a source for instability of decision tree algorithms in high dimensional datasets." *Artificial Intelligence Review* 43.2 (2015): 301-310.

[98] Cadenas, José M., M. Carmen Garrido, and Raquel Martínez. "Fuzzy discretization process from small datasets." *Computational Intelligence*. Springer International Publishing, 2016. 263-279.

[99] Breiman, Leo. "Heuristics of instability and stabilization in model selection." *The annals of statistics* 24.6 (1996): 2350-2383.

[100] http://scikit-learn.org/stable/auto_examples/svm/plot_svm_kernels.html

[101] http://scikit-learn.org/stable/modules/svm.html

[102] http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[103] Hussain, Zanab, John Paul Mueller, and Luca Massaron. *Python for Data Science for Dummies*. John Wiley & Sons, 2015.

[104] http://scikit-learn.org/stable/modules/svm.html

[105] Rennie, Jason D., et al. "Tackling the poor assumptions of naive bayes text classifiers." *ICML*. Vol. 3. 2003.

4.2.4.6 The Naïve Bayes

Naïve Bayes classification system is one of the most widely used classification systems, especially in the academia. The term *Naive* represents the simplicity of the system because this considers all features in an instance as individual without depending on any other feature. Such consideration lets us do the calculation without much overload. The Naïve Bayes system finds the overall or posterior probability for each instance in each class and compares the results in each class and the class with better probability is attached to the given sample. In SK Learn Multinomial NB, there is a smoothing parameter called *alpha* which assigns a small, non-zero probability to rare or non-existent terms and the classification probability would not become zero only because of non-existent terms. For the best alpha value, we performed grid search on base ten log scale, specifically from 10e-3 – 10e6. Logarithmic scale is recommended for hyperparameters such as alpha because then we can search faster in a bigger space.

4.2.4.7 The Linear SVM

The SVM has long been favorited by many researchers over the years and is one of the most accurate classifiers in terms of learning and predicting correctly. The SVM is a supervised classifying algorithm which tries to separate the feature space into different portions depending on the closest points (or support vectors which are our training folktales) to the separating line. This separating boundary is a line in 2D (represented by a dot in 1D, a plane in 3D and a hyperplane in xD: D-dimension) and these lines partition *fruitarian* group from *vegetarian* or other food groups in our case. The SVM tries to find the optimal line and to keep the separation margins as wide as possible as in Figure 10.

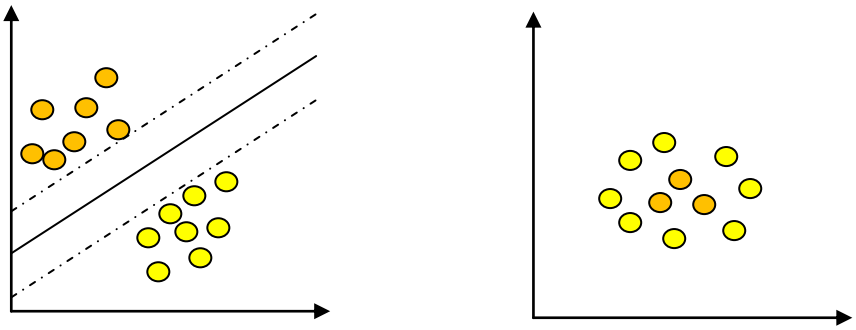

Figure 10. Linearly separable (on the left) vs. non-linear data

For nonlinear problems, the separating line becomes more curved or warped, the algorithm tends to overfit and spends more time yet with not more significant results. Therefore, it is generally recommended to use the simpler linearly separable algorithm whenever possible and use nonlinear kernels only when the data is not linearly separable because training the

kernel classifiers on text documents was already known to be a waste of time and efficiency[106].

With SVM in Scikit Learn, we can tweak with four parameters: the kernel, C, gamma and degree parameters. We learned that text classification is mostly linear, so we choose the kernel parameter to be linear. With the linear kernel, we do not need to change gamma and degree parameters. So we only care about the parameter C in our case.[111]

The C parameter is about how much our classification model has to learn from the training points (each training point is a folktale). When the parameter is small, the SVM generalizes more and uses only a few of the points, when it's high the SVM is forced to learn by following more training points. The higher C is, the model tends to overfit, which means the SVM learns too much about training points and is not able to handle new data well while too small C tends to overgeneralize missing out on certain outliers and so on. By tuning and trying, we find the middle C that would be optimal for our classifier. We need to find the optimal separator with the biggest margin on all sides around it when it separates different class points. For the best C amount, we performed grid search on base ten log scale, specifically from 10e-3 – 10e6; lower ranges are too costly and not used often in practice.

Linear SVMs use a series of binary classifiers to deal with multiclass classifications as they are inherently binary. The two of the SVM algorithms in Scikit Learn are linear: SVC with *linear* kernel and LinearSVC. And they use distinct series of binary classifiers for multiclass problems.

SVC with *linear* kernel is LibSVM[107] based and uses One Vs One (**OVO**) reduction for multiclass classification. This creates k(k−1)2 classifiers and each one is trained on data from two different classes. When predicting, it selects the class that gets the more votes. Because its learning process involves only two classes at a time, this performs better for **unbalanced** data than OVR. OVO method was shown to be more effective than other multiclass methods[108]. OVO approach was shown to have better performance for noisy dataset as well[109].

LinearSVC is LibLinear based and uses One Vs Rest (**OVR**) approach to deal with multiclass classification[110]. This algorithm is much faster and good for high dimension of both instances and features[111]. However, the OVR approach compares all classes at a time when predicting, so works better for **balanced** data as this requires the complete dataset k times[112] and this can be computationally expensive on low performance machines.

---

[106] https://www.csie.ntu.edu.tw/~cjlin/papers/kernel-check/kcheck.pdf

[107] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[108] Hsu, Chih-Wei, and Chih-Jen Lin. "A comparison of methods for multiclass support vector machines." *IEEE transactions on Neural Networks* 13.2 (2002): 415-425.

[109] Sáez, José A., et al. "Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition." *Knowledge and information systems* 38.1 (2014): 179-206.

[110] http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[111] Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." (2003): 1-16.

[112] http://scikit-learn.org/stable/modules/multiclass.html#one-vs-one

As the SK Learn Linear SVC algorithm has an optional parameter to test both OVO and OVR, we kept checking both options routinely.

4.2.4.8 Dataset split

With machine learning, the machine needs to learn from the training set before it gets tested on the test set. So, we need to split our corpus as an entire dataset into a portion the machine can learn from and a portion the machine can test on. It is a recommended practice to split the dataset into three portions, namely, training, validation and test sets[113]. The training set is used to fit the learning data, the validation set is used to fine tune the hyperparameters of the model and the test set is used only once to extract a tangible evaluation results. But for model selection, we need the only training and validation sets since we aim to choose the best model without needing to see how our selected model would do in the real world. We already learned that the best model for our problem is the Linear SVM from our theoretical background; however, we still need to check one more linear algorithm, NB, to compare. For this comparison, we need to split our dataset into training, validation and test sets and see if our theory would be confirmed on our skewed dataset. We store the test set separately for the final evaluation run.

After selecting the model, we will augment our data and balance out all the classes. The augmentation will be implemented separately on the training and test sets because they have their own distinct features. For small size datasets, one of the validation and test sets needs to be sacrificed as splitting into three separate sections is not practical for a small dataset. Therefore, we will train and validate on the training set using the cross validation, use the best parameter option from the best model found during model selection and perform the final evaluation on the test set. After the test set performance, we can make our final conclusion comparing with the rule-based classifier.

4.2.4.9 Cross validation for normalized metrics

The traditional train, validation and test split would suffice when you have a large enough dataset. For small datasets, though, we may not encounter with all possible features in one training session. We need to make sure our model sees most every feature in the training set as we do not have a separate validation set. K fold cross validation comes into light so we can divide our training set into k equal sets and use the k-1 parts for training and the other one part for validation with k repetitions. The number of folds is usually ten in practice. After the K repetitions, our model has now learned from all features and we get more balanced and averaged metrics as a result and choose the model with the best overall performance.

4.2.4.10 Stratified allocation

Once we split up our randomized data into train and test sets on our highly skewed dataset, we can easily foresee that the data would be even more one sided for any one or so sets out of the two. It is possible that all of our *fruitarian* folktales taken up by the test set and there would be no presence of this class in the training set and the whole model would fail. Now

[113] Ng, Andrew. "Lecture 61: Model Selection and Training Validation Test Sets." Machine Learning, Coursera. Web. 11 Jul. 2017.

that we understand the danger of random sampling, we see why we needed the stratification. The stratification allows us to place equal percentage of folktales from each class to the split sets. If five percent of the whole corpus is *fruitarian* folktales, then five percent of training, validation or test sets should contain fruitarian data points. In our case, the following amount belonged to each class after stratification, F: 33, N: 253, O: 297, V: 154, Vg, 44 where each of the ten cross validation and one test folds contain equal amount from each class while the initial corpus of 804 tales consisted of unequal amounts, F: 33, N: 262, O: 302, V: 163, Vg, 44. Once we separate the stratified test set of 71 tales (F: 3, N: 23, O: 27, V: 14, Vg: 4), the stratified train validation set consists of 710 tales of 10 equal parts (F: 30, N: 230, O: 270, V: 140, Vg, 40)

It is another way of improving the dataset skewness and handling overfitting, however this only rearranges the dataset into sequence of k folds where each fold contains equal percent data from each class and our dataset still stays skewed unless we balance out it using the Virtual Example Oversampling method.

4.2.4.11 Oversampling based on Virtual Examples

Although we tried to solve our problem tuning the parameters as much as possible to improve the performance of our system, the standard machine learning algorithms tend to bias towards the majority classes[114] and the resulting scorings were not improving much. Therefore, in an alternative way to handle the unbalanced class issue, we need to collect more data to balance out the dataset. However, collecting more data is time consuming and laborious; fortunately, certain researchers came up with a brilliant idea to balance out data. Depending on the nature of the problem, the sampling method either removes instances from the overrepresented class (undersampling) or adds copies of instances from the underrepresented class (oversampling). Because we do not have many instances and cannot delete from the majority class, we choose the oversampling way. There could be different ways to realize oversampling such as Virtual Examples (VE) or Synthetic Minority Oversampling technique (SMOTE). The VE create virtual examples from the support vectors for text classification assuming the document class is unchanged after adding a small number of words added or deleted. This has been tested and improved the performance of text classification with SVMs, especially for small training sets[115]. Similar to VE, SMOTE randomly samples the attributes from instances in the minority class.

---

[114] Fernández, Alberto, Mara José Del Jesus, and Francisco Herrera. "Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning." *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Springer Berlin Heidelberg, 2010.

[115] Sassano, Manabu. "Virtual examples for text classification with support vector machines." *Proceedings of the 2003 conference on Empirical methods in natural language processing*. Association for Computational Linguistics, 2003.

Figure 11. Hyperplane and Virtual examples

In this method, we can add or delete a small number of words without changing the document category following the VE method. We applied general a cleanup process in advance such as tokenizing, removing punctuations and stop words and so on.

We fed our imbalanced data to a support vector classifier to extract the support vectors. After taking the support vectors out with the respective classes, we calculated how many more samples are needed for each underrepresented class. The process of creating virtual examples based on the support vectors are simple. We simply select a random support vector and to it, we added one word by randomly selecting from the class specific food vocabulary.

After stratification and separating the test set for later evaluation, we have 710 (F: 30, N: 230, O: 270, V: 140, Vg, 40) tales for training and validation. The class with the highest amount is O-Omnivorous with 270 samples; so, all the other underrepresented class should fill additional tales up to this number using the virtual examples and the final training-validation set would contain 1350 tales (270 multiplied by each of the five classes) once augmented. The testing set of 71 tales (F: 3, N: 23, O: 27, V: 14, Vg, 4) will become 135 after augmentation in the same way. The augmentation process involves taking a randomly selected support vector suitable for the augmenting class and adding one unambiguous food term or combination of eat verb followed by an ambiguous food term that are randomly selected depending on the augmenting class and then randomly injects them into any given tale text required for the process. For example, if the support vector belonged to the class *vegan* then we could inject just any one random vegan term such as *potato* because all vegan food terms are unambiguous. If the support vector was omnivorous then we could randomly inject one unambiguous flesh term such as *beef* or one eat verb such as *roast* followed by a random ambiguous flesh term such as *sheep* into the vector and we get one more newly created tale.

5 Results/Findings

5.1 *Results from Rule-based Classification*

The rule-based classifier is not needed to be trained. Instead, it follows specific hard coded rules to reach the final decision. Its scores based on the training set (that is divided into training and validation set; in machine learning, this division was repeated ten times and called cross validation) and test set combined is as following:

Scores	Train and test sets combined
Accuracy	**0.8445**
Precision	**0.8491**
Recall	**0.8445**
F1	**0.8468**
confusion matrix	F N O V Vg F [[28 1 2 2 0] N [10 226 15 10 1] O [7 19 254 16 6] V [0 1 19 139 4] Vg [1 4 7 0 32]]
Classification report	precision recall f1-score support f 0.61 0.85 0.71 33 n 0.90 0.86 0.88 262 o 0.86 0.84 0.85 302 v 0.83 0.85 0.84 163 vg 0.74 0.73 0.74 44 avg / total 0.85 0.84 0.85 804

Table 11. Rule-based performance on the whole dataset

The scores above is relatively high because some of the features in the training set are already seen. So we need to run our algorithm on the unseen features in the test set only. And the corresponding scores are as following:

Sscores	Test set with unseen features
Accuracy	**0.8028**
Precision	**0.8035**
Recall	**0.8028**
F1	**0.8032**
confusion matrix	F N O V Vg F [[3 0 0 0 0] N [1 19 2 1 0] O [1 2 21 2 1] V [0 0 0 13 1] Vg [0 1 2 0 1]]
Classification report	precision recall f1-score support f 0.60 1.00 0.75 3 n 0.86 0.83 0.84 23 o 0.84 0.78 0.81 27 v 0.81 0.93 0.87 14 vg 0.33 0.25 0.29 4 avg / total 0.80 0.80 0.80 71

Table 12. Rule-based performance on test set with unseen features

5.2 *Results from Computational Classifier*

To be able to compare the results from the classifiers, first we needed a baseline dummy classifier, a.k.a Zero R classifier, that assigns the class of the most common class to all the predicted folktales. The most commonly occurred classes were *omnivorous* in our corpus.

Due to the imbalanced nature of the dataset in our computational experiment, the micro averaged scores were all identical and macro averaged scores were really poor. Therefore, we pay attention to only the weighted averaged scores. Please refer to Table 13 for the baseline results.

	ZeroR Baseline		
Averaged scores	Micro	Weighted	Macro
Accuracy	0.3803		
Precision	0.3803	0.1446	0.0761
Recall	0.3803	0.3803	0.1999
F1 score	0.3803	0.2095	0.1102
Confusion matrix	F N O V Vg F [[0 0 30 0 0] N [0 0 230 0 0] O [0 0 270 0 0] V [0 0 140 0 0] Vg [0 0 40 0 0]]		

Table 13. Baseline dummy classifier results

The confusion matrix were created for each classification and the classes are *fruitarian* as *F*, *neutral* as *N*, *omnivorous* as *O*, *vegan* as *V* and *vegetarian* as *Vg*, in the alphabetical order. We can see the dummy classifier classified everything as omnivorous after the ten-fold cross validation.

5.2.1 Feature Selection

Then the first experiment done with Naïve Bayes with its default settings (defaults settings do not usually do well with problematic datasets but here we are just trying to choose better features for the sake of simplicity) and unigram and all the terms in a tale except those cleaned out, such as the stop words and punctuations, to see how the most naïve classifier would do with all the features with no filtration. (See Table 14)

Averaged scores	NB alpha=1.0, default, unigram		
	Micro	Weighted	Macro
Accuracy	0.3803		
Precision	0.3803	0.1446	0.0761
Recall	0.3803	0.3803	0.1999
F1 score	0.3803	0.2095	0.1102
	F N O V Vg F [[0 0 30 0 0] N [0 0 230 0 0] O [0 0 270 0 0] V [0 0 140 0 0] Vg [0 0 40 0 0]]		

Table 14. Naïve Bayes results with default settings and all terms

This first attempt with NB was no better than the baseline classifier. And as we expected with the noisy, small and imbalanced dataset, this simplest form of machine classification was not learning well. The confusion matrix indicates that the classifier did not assign any label properly. Then we decided to take only the nouns and verbs as those are the more specific information we need as features. Please see Table 15 for the results.

Averaged scores	NB (alpha=1.0, unigram)		
	Micro	Weighted	Macro
Accuracy	0.3873		
Precision	0.3873	0.2385	0.1338
Recall	0.3873	0.3873	0.2043
F1 score	0.3873	0.2789	0.1510
	F N O V Vg F [[0 1 29 0 0] N [0 5 225 0 0] O [0 0 270 0 0] V [0 3 137 0 0] Vg [0 1 39 0 0]]		

Table 15. Naïve Bayes results with default settings, only nouns and verbs as features.

With all nouns and verbs as features, the scores were slightly improved: the accuracy 0.3873 and the f1 score 0.2789. The weighted averaging was used for the previously mentioned reasons. However, the machine was still confused among many unrelated nouns and verbs and did not know how to divide into different categories as our food related terms were really scarce in each tale.

For the next step, we filtered more and picked only the food related nouns and verbs we are interested in. See Table 16.

Averaged scores	NB alpha=1.0, default, unigram		
	Micro	Weighted	Macro
Accuracy	0.4859		
Precision	0.4859	0.4723	0.3361
Recall	0.4859	0.4859	0.3043
F1 score	0.4859	0.4787	0.3189
	F N O V Vg F [[0 8 22 0 0] N [0 53 170 7 0] O [0 28 231 11 0] V [0 25 54 61 0] Vg [0 1 37 2 0]]		

Table 16. Naïve Bayes results with default settings, related nouns and verbs as features.

With this last experiment in Table 16, we get relatively much improved scores (accuracy = 0.4859, f1=0.4787) compared with the previous results. For instance, the confusion matrix shows that the classifier properly handled 61 vegan classes while there was no vegan class found with the previous attempt.

NB (alpha=1, unigram) ten fold cross validation classification reports

Fold 1

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.40	0.09	0.14	23
o	0.45	0.89	0.60	27
v	0.69	0.64	0.67	14
vg	0.00	0.00	0.00	4
avg / total	0.44	0.49	0.41	71

Fold 2

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.67	0.26	0.38	23
o	0.45	0.89	0.60	27
v	0.78	0.50	0.61	14
vg	0.00	0.00	0.00	4
avg / total	0.54	0.52	0.47	71

Fold 3

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.46	0.26	0.33	23
o	0.41	0.78	0.54	27
v	0.71	0.36	0.48	14
vg	0.00	0.00	0.00	4
avg / total	0.45	0.45	0.41	71

Fold 4

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.40	0.26	0.32	23
o	0.44	0.78	0.56	27
v	0.75	0.43	0.55	14
vg	0.00	0.00	0.00	4
avg / total	0.44	0.46	0.42	71

Fold 5

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.33	0.13	0.19	23
o	0.44	0.93	0.60	27
v	1.00	0.36	0.53	14
vg	0.00	0.00	0.00	4
avg / total	0.47	0.46	0.39	71

Fold 6

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.46	0.26	0.33	23
o	0.42	0.81	0.56	27
v	0.67	0.29	0.40	14
vg	0.00	0.00	0.00	4
avg / total	0.44	0.45	0.40	71

Fold 7

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3

Fold 8

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3

Fold 9

	precision	recall	f1-score	support
f	0.00	0.00	0.00	3

n	0.47	0.30	0.37	23	n	0.50	0.22	0.30	23	n	0.50	0.26	0.34	23
o	0.47	0.85	0.61	27	o	0.41	0.81	0.54	27	o	0.51	0.93	0.66	27
v	0.86	0.43	0.57	14	v	0.86	0.43	0.57	14	v	0.70	0.50	0.58	14
vg	0.00	0.00	0.00	4	vg	0.00	0.00	0.00	4	vg	0.00	0.00	0.00	4
avg / total	0.50	0.51	0.46	71	avg / total	0.49	0.46	0.42	71	avg / total	0.49	0.54	0.48	71

Fold 10

precision	recall	f1-score	support
f	0.00	0.00	0.00	3
n	0.43	0.26	0.32	23
o	0.50	0.89	0.64	27
v	0.67	0.43	0.52	14
vg	0.00	0.00	0.00	4
avg / total	0.46	0.51	0.45	71

Table 17. Classification reports from the NB (alpha=1, unigram) with related nouns and verbs filtered

When we look at the classification report results from Table 17, the underrepresented classes *f* and *vg* always get zero scores compared with the more frequent classes *o, n* and *v* with more training samples. And this confirms that we indeed had a class imbalance problem.

As can be seen from our experiments with the simplest classifier with default settings and its results with different feature sets, the machine algorithm was learning the best when the features were the food related noun terms and eating related verb terms filtered out whereas it was not learning anything when there was no filtering in the feature set. As a result, we set to proceed with the related nouns and verbs as our features for the experimental models with more fine tuned parameter settings.

5.2.2 Choosing the Classifier

With our features defined, now we need to choose the one that performs better out of the two linear classification algorithms.

We grid searched the two linear classifiers, namely, Multinomial NB and Linear SV tuning the hyperparameters alpha (*See The Naïve Bayes 4.2.4.6*) and C (*See The Linear SVM 4.2.4.7*) respectively with uni-tri grams to get the best possible scores from each one of them. The alpha and C value range was ten base log 1e-3 – 1e6.

We used bag of words and ten-fold cross validation against the ZeroR Baseline classifier to see which classifier gets better results.

5.2.2.1 Naïve Bayes Performance on Imbalanced Dataset

N-gram	Averaged scores	Multiclass Classification, optimal alpha=0.001		
		Micro	Weighted	Macro
Unigram	Accuracy	0.5408		
	Precision	0.5408	0.5553	0.5102
	Recall	0.5408	0.5408	0.3963
	F1	0.5408	0.5476	0.4442
	Aggregated confusion matrix	F N O V Vg F [[1 4 20 5 0] N [1 55 166 8 0] O[2 21 231 14 2] V [1 17 33 88 1] Vg [0 1 29 1 9]]		
Bigram		NB, optimal alpha=0.01		
	Accuracy	**0.5437**		
	Precision	0.5437	**0.5828**	0.5649
	Recall	0.5437	**0.5437**	0.4471

	F1	0.5437	**0.5620**	0.4978
	Aggregated confusion matrix	F N O V Vg F [[6 4 11 7 2] N [1 53 167 9 0] O [3 16 227 21 3] V [5 7 37 86 5] Vg[0 2 20 4 14]]		
Trigram		NB, optimal alpha=0.01		
	Accuracy	0.5268		
	Precision	0.5268	0.5493	0.5328
	Recall	0.5268	0.5268	0.4310
	F1	0.5268	0.5373	0.4751
	Aggregated confusion matrix	F N O V Vg F [[6 5 11 7 1] N [1 57 163 9 0] O [6 21 217 23 3] V [4 11 40 81 4] Vg [0 2 21 4 13]]		

Table 18. Naïve Bayes classifier with advanced parameter and ngram settings

After grid searching the NB model, its best performance was revealed from the bigram feature extraction and optimal alpha equals 0.01 (accuracy=0.5437 and f1=0.5620)

5.2.2.2 Linear SVM Performance on Imbalanced Dataset

We add more parameters and tunings and see if the scores improve with the SVM algorithm.

Averaged scores	ZeroR Baseline			**SVM, Unigram, Optimal C=10**					
				OVO			OVR		
	Micro	Weighted	Macro	Micro	**Wtd.**	Mac.	Micro	Wtd.	Macro
Accuracy	0.3803			**0.7155**			0.7197		
Precision	0.3803	0.1446	0.0761	0.7155	**0.7377**	0.6341	0.7155	0.7377	0.6341
Recall	0.3803	0.3803	0.1999	0.7155	**0.7155**	0.6357	0.7155	0.7155	0.6357
F1	0.3803	0.2095	0.1102	0.7155	**0.7262**	0.6290	0.7155	0.7262	0.6290
Aggregated confusion matrix	F N O V Vg F [[0 0 30 0 0] N [0 0 230 0 0] O [0 0 270 0 0] V [0 0 140 0 0] Vg [0 0 40 0 0]]			F N O V Vg F [[12 10 4 3 1] N [13 195 14 7 1] O [15 34 178 34 9] V [8 17 11 101 3] Vg [1 3 10 4 22]]					
	SVM, Bigram, Optimal C=10						SVM, Trigram, Optimal C=1					
Averaged scores	OVO			OVR			OVO			OVR		
	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.
Accuracy	0.6986			0.6986			0.7000			0.7000		
Precision	0.6986	0.6958	0.6015	0.6986	0.6958	0.6015	0.7000	0.7122	0.6285	0.7000	0.7122	0.6285
Recall	0.6986	0.6986	0.5673	0.6986	0.6986	0.5673	0.7000	0.7000	0.6194	0.7000	0.7000	0.6194
F1	0.6986	0.6971	0.5827	0.6986	0.6971	0.5827	0.7000	0.7020	0.6234	0.7000	0.7020	0.6234
Aggregated confusion matrix	F N O V Vg F [[9 14 5 2 0] N [3 205 14 8 0] O [12 42 180 29 7] V [6 18 26 88 2] Vg[0 4 13 9 14]]						F N O V Vg F [[11 14 5 0 0] N [6 194 23 7 0] O [15 42 172 34 7] V [10 16 15 98 1] Vg [1 4 6 7 22]]					

Table 19. SVM model selection with advanced parameter and ngram settings on imbalanced dataset

The best SVM performance was observed when its settings were unigram and optimal C equals 10 (accuracy=0.7155, f1=0.7262).

5.2.2.3 The Selected Classifier

As expected, the SVM results were higher than those from the NB model. The NB is a very simple and naive classifier, it does not handle unbalanced dataset well while the SVM naturally handles text classification problems very well (See Classifier Choice 4.2.4.5).

Throughout all the experiments up until now, the macro averaged results were always chaotic and micro averaged scores were the same due to highly skewed data, so we can check only the weighted averaged scores. We can see more properly classified data in the confusion matrices of both models with advanced parameter settings.

The decision function shapes OVR and OVO made no difference in any of the experiments.

After stratification and separating the hold out set for testing, our model comparison ran on total 710 training tales out of the corpus of 804 tales. We conducted a Cochran's Q statistical test among the Baseline Zero R model, the best performing NB and SVM models. As a result, there was a statistically significant difference χ^2 *(2, N=710)=190.12, p-value<0.05* among the percentage of correctly labeled folktales in Baseline (38%), SVM (71.5%) and NB models (54.4%). Furthermore, we conducted McNemar's pairwise test for each pair of models and we conclude that there was a strong significant differences between the percentages of folktales that were correctly classified in the Baseline model and SVM χ^2 *(1, N=710)=133.7, p-value<0.05* followed by the significant difference of the pairwise comparison between Baseline model and NB χ^2 *(1, N=710)=66, p-value<0.05* with the last significant difference between SVM and NB χ^2 *(1, N=710)=54.7, p-value<0.05*. Therefore, the differences among the models were not by chance and we choose the model with the best performance, which is the SVM as we predicted theoretically.

The results let us safely proceed with the SVM without doubt.

5.2.3 Cross-Validated scores with Balanced (Augmented) Classes

After augmenting our dataset with virtual examples based on the support vectors extracted from the best model performance on the imbalanced dataset, we have a balanced dataset finally. Since we have a balanced out dataset, we can expect the model to perform better and all averaging methods would get better. Also, we can use the accuracy as our main metric instead of f1 score. The accuracy is a simple probability of 1 / k (k=number of classes) as we now have equal amount of samples in each class and this calculation is called Random Baseline Accuracy.

With the same parameter tweaking settings with the previous SVM selection procedure, the best performance was observed when the settings were trigram and optimal C equals 1 (See Table 20). Ten fold cross validation was used as before. The scores improved on the balanced dataset (accuracy = 0.8792, f1=0.8812) compared with those on the unbalanced dataset (accuracy=0.7155, f1=0.7262) and the Random Baseline Accuracy (accuracy = 1/5= 0.20)

Averaged scores	SVM, Unigram, Optimal C=10						SVM, Bigram, Optimal C=10					
	OVO			OVR			OVO			OVR		
	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.	Mic.	Wtd.	Mac.
Accuracy	0.8696			0.8696			0.8778			0.8778		
Precision	0.8696	0.8748	0.8748	0.8696	0.8748	0.8748	0.8778	0.8827	0.8827	0.8778	0.8827	0.8827
Recall	0.8696	0.8696	0.8696	0.8696	0.8696	0.8696	0.8778	0.8778	0.8778	0.8778	0.8778	0.8778
F1	0.8696	0.8722	0.8722	0.8696	0.8722	0.8722	0.8778	0.8802	0.8802	0.8778	0.8802	0.8802
Aggregated confusion matrix	F N O V Vg F [[256 12 0 2 0] N [6 240 15 8 1] O [8 37 187 28 10] V [3 16 20 231 0] Vg [1 5 2 2 260]]						F N O V Vg F [[261 5 2 2 0] N [3 245 13 8 1] O [9 41 179 33 8] V [1 17 15 237 0] Vg [0 6 0 1 263]]					

Averaged scores	**SVM, Trigram, Optimal C=1**					
	OVO			OVR		
	Mic.	**Wtd.**	Mac.	Mic.	Wtd.	Mac.
Accuracy	**0.8792**			0.8792		
Precision	0.8792	**0.8831**	0.8831	0.8792	0.8831	0.8831
Recall	0.8792	**0.8792**	0.8792	0.8792	0.8792	0.8792
F1	0.8792	**0.8812**	0.8812	0.8792	0.8812	0.8812
Aggregated confusion matrix	F N O V Vg F [[259 7 3 1 0] N [3 229 28 9 1] O [7 34 191 30 8] V [0 16 9 244 0] Vg [0 4 1 1 264]]					

Table 20. SVM model selection with advanced parameter and ngram settings on balanced dataset

Finally, on the best chosen model, we performed a training and test split. We had divided the corpus into eleven equal portions and we used the first ten folds for training and validation, and the last one was reserved for testing only. Both the training-validation and the reserved testing portions were augmented separately using the general and portion specific features so that the reserved testing portion will be kept unseen to the machine classifier.

Scores	Hold-Out set with its own feature set
	OVO, Random BL acc=0.2
	Weighted, c=10, bigram
Accuracy	**0.7630**
Precision	**0.7786**
Recall	**0.7630**
F1	**0.7707**
confusion matrix	F N O V Vg F [[27 0 0 0 0] N [1 19 6 1 0] O [1 3 20 2 1] V [1 12 1 13 0] Vg [0 2 0 1 24]]
	precision recall f1-score support f 0.90 1.00 0.95 27 n 0.53 0.70 0.60 27 o 0.74 0.74 0.74 27 v 0.76 0.48 0.59 27 vg 0.96 0.89 0.92 27 avg / total 0.78 0.76 0.76 135

Table 21. SVM performance on unseen data

Following the hierarchical classification rules from the rule-based classifier, we tested a similar structure consisting of four binary classifiers, the best SVM ones, on the balanced data. Each of the four classifiers is trained on one specific class against the others (1. O vs.

Vg+V+F+N, 2. Vg vs. V+F+N, 3. V vs. F+N, 4. F vs. N) and the final classification assigns the class with the best probability.

	Hold-Out set with its own feature set		Hold-Out set with its own feature set		Hold-Out set with its own feature set	
scores	**OVO, Random BL acc=0.2**		OVO, Random BL acc=0.2		OVO, Random BL acc=0.2	
	Weighted, c=1(ovg), 10(vgv), 1(vf), 1(fn), unigram		Weighted, c=100(ovg), 1 (vgv), 10 (vf), 1 (fn), bigram		Weighted, c=10 (ovg), 1 (vgv), 10 (vf), 1 (fn), trigram	
Accuracy	**0.6889**		0.6519		0.6815	
Precision	**0.8587**		0.8107		0.8230	
Recall	**0.6889**		0.6519		0.6815	
F1	**0.7645**		0.7226		0.7456	
confusion matrix	O VgVFN O [[212 58] VgVFN [82 998]] VFN Vg VFN [[808 2] Vg [6 264]]	F V F [[526 14] V[8 262]] F N F [[261 9] N [10 260]]	O VgVFN O [[196 74] VgVFN [39 1041]] VFN Vg VFN [[807 3] Vg [5 265]]	F V F [[530 10] V [10 260]] F N F [[261 9] N [6 264]]	O VgVFN O [[201 69] VgVFN [41 1039]] VFN Vg VFN [[807 3] Vg [3 267]]	F V F [[531 9] V [10 260]] F N F [[263 7] N [6 264]]

Table 22. Multiple binary classifiers on unseen data

The final result with the accuracy of 0.6889, however, is not better than the result of one multiclass classifier with accuracy of 0.7630.

Our final result from the hybrid machine learning classification on the unseen data was improved compared with the baseline accuracy (0.7630 > 0.20) and we yet have to compare it with the result from the rule-based classifier.

6 Conclusion

We performed two proportion independent Z test on the final two best models. The selected rule-based model was based on numerous rules to make a final decision, but we did not need to stratify and balance the corpus. The selected SVM was based on both rules and machine learning techniques for the final decision, and its stratified and augmented corpus was entirely different and randomized independently. The null hypothesis was the proportional results of correct instances were similar. The alternative hypothesis was simply they were different. The result in the number of correctly labeled tales between the rule-based and SVM models indicated that it was not statistically significant (z= -0.65, p>0.05). Hence, we are not able to reject the null hypothesis and we can conclude that both the rule-based model (accuracy=0.8028, n=71) and the SVM model with augmented data points (accuracy=0.7630, n=135) would likely to perform equally competitive with unseen tales.

After all, we have achieved relatively successful scores with our classification algorithms and therefore folktales can be classified into different dietary categories with careful feature and data processing. Now, those on alternative diets could enjoy old stories without food related concerns to a certain extent.

7 Deployment

The best classifier, the rule-based classifier, is deployed online at:
http://folktaleclassification.pythonanywhere.com/
And the best performing SVM demo can be found online at:
http://colorvori.pythonanywhere.com/
This was the only free host that can handle Python codes and machine learning algorithms well and easily using Flask application web interface. Please use a folktale saved in .TXT format.

8 Discussion

First major issue was lack of balanced data points. We tried to fix the class imbalancing issue by tweaking certain parameters and oversampling with virtual examples. As a result, we got 0.7852 percent accuracy on the augmented test data. But nothing can replace the real data. Our corpus size was also small. Collecting more samples and building a bigger corpus may solve these issues.

One of the main issues during the project was the question of handling the anaphoric instances. Although we adopted the well-known coreference resolution tool from the Stanford team, it correctly found only the 13.61% antecedents out of the 147 anaphoric instances. There could be a better performing tool that can solve this problem.

Perhaps, it would be clearer for the audience if we detect cannibalism as a separate case of omnivorous. Currently, they are both mixed up. Also, we excluded all kinds of drinks such wine (made out of fruits) and beer (made out of grains). They can be included in the food items. Furthermore, our work is able to handle certain bread related idiomatic expressions such as *earn your bread, win his daily bread, beg for bread* and *daily bread* at the moment and it could to be expanded to include more diverse food groups.

Both classifiers can include an extended module where new features can be added from new tales. Currently, the classifiers will make a mistake when faced with a new term such as a *troll* from Norwegian tales.

Finally, we could extend the corpus with more folktales from more diverse nationalities too.

9 References

Abello, James, Peter Broadwell, and Timothy R. Tangherlini. "Computational folkloristics." *Communications of the ACM* 55.7 (2012): 60-70.

Aggarwal, Charu C., and ChengXiang Zhai, eds. Mining text data. Springer Science & Business Media, 2012, pp. 200.

Alcantud Díaz, María. "Violence in the brothers Grimm's Fairy Tales: a Corpus-Based Approach." *Revista Alicantina de Estudios Ingleses, 2010, p. 173-185* (2010).

Andrievskikh, Natalia. "Food Symbolism, Sexuality, and Gender Identity in Fairy Tales and Modern Women's Bestsellers." Studies in Popular Culture 37.1 (2014): 137-153.

Bird, Steven, Ewan Klein, and Edward Loper. Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.", 2009

Broadwell, Peter M., David Mimno, Timothy R. Tangherlini. "The Tell-Tale Hat: Surfacing the Uncertainty in Folklore Classification." Journal of Cultural Analytics (2017).

Brownlee, Jason. "Overfitting and Underfitting With Machine Learning Algorithms". N.p., 21 Mar. 2016. Web. 10 July. http://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/

Cadenas, José M., M. Carmen Garrido, and Raquel Martínez. "Fuzzy discretization process from small datasets." Computational Intelligence. Springer International Publishing, 2016. 263-279.

Canonici, Noverino Noemio. "Food in Zulu folktales." Southern African Journal for Folklore Studies 2.1 (1991): 24-36.

Cutts, Martin. Oxford guide to plain English. OUP Oxford, 2013.

Declerck, Thierry, and Lisa Schäfer. "Porting past Classification Schemes for Narratives to a Linked Data Framework." Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage. ACM, 2017.

Declerck, Thierry, Tyler Klement, Antonia Kostova. "Towards a WordNet based Classification of Actors in Folktales." Software Project Course *Classification of Folktales*. University of Saarland, 2015.

Flanagan, Michael. "Cowpie, Gruel and Midnight Feasts: the representation of Food in Popular Children's Literature." (2012).

Groza, Adrian, and Lidia Corde. "Information retrieval in falktales using natural language processing." *Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on*. IEEE, 2015.

Harikrishna, D. M., and K. Sreenivasa Rao. "Children story classification based on structure of the story." Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on. IEEE, 2015.

Harun, Harryizman, and Zulikha Jamaludin. "Folktale conceptual model based on folktale classification system of type, motif, and function." Proceeding of the 4th international conference on computing and informatics (ICOCI). 2013.

Honeyman, Susan. "Gingerbread Wishes and Candy (land) Dreams: The Lure of Food in Cautionary Tales of Consumpion." Marvels & Tales 21.2 (2007): 195-215.

Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." (2003): 1-16.

Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." Machine learning: ECML-98 (1998): 137-142.

Meder, Theo, et al. "Automatic enrichment and classification of Folktales in the Dutch Folktale Database." Journal of American Folklore 129.511 (2016): 78-96.

Mosley, Lawrence. "A balanced approach to the multi-class imbalance problem." (2013).

Ng, Andrew. "Lecture 61: Model Selection and Training Validation Test Sets." Machine Learning, Coursera. Web. 11 Jul. 2017.

Nguyen, Dong, Dolf Trieschnigg, and Mariët Theune. "Folktale classification using learning to rank." European Conference on Information Retrieval. Springer Berlin Heidelberg, 2013.

Nguyen, Dong-Phuong, et al. "Automatic classification of folk narrative genres." (2012).

Nguyen, Quynh C., et al. "Leveraging geotagged Twitter data to examine neighborhood happiness, diet, and physical activity." Applied Geography 73 (2016): 77-88.

Piper, Brenda. Diet and nutrition: a guide for students and practitioners. Springer, 2013.

Propp, Vladimir. "Morphology of the Folktale, trans." Louis Wagner, 2d. ed.(1928 (1968).

Roosman, Raden S. "Coconut, breadfruit and taro in Pacific oral literature." The Journal of the Polynesian Society 79.2 (1970): 219-232.

Sassano, Manabu. "Virtual examples for text classification with support vector machines." Proceedings of the 2003 conference on Empirical methods in natural language processing. Association for Computational Linguistics, 2003.

Seki, Keigo. "Types of Japanese folktales." *Asian Folklore Studies* 25 (1966): 1-220.

Sokolova, Marina, and Guy Lapalme. "A systematic analysis of performance measures for classification tasks." Information Processing & Management 45.4 (2009): 427-437.

Spruyt, Vincent. "The Curse of Dimensionality in classification." N.p., 16 Apr. 2014. Web. 15 June 2017. http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/.

Weingart, Scott, and Jeana Jorgensen. "Computational analysis of the body in European fairy tales." *Literary and Linguistic Computing* 28.3 (2012): 404-416.

10 Appendices

10.1 *Appendix A: Latin-1 Supplement Character Set with Replacement Equivalents*

Unicode #	Latin-1 supplement character set		Replacement character set	
128, 162, 163, 165	euro, cent, pound, yen signs	€, ¢, £, ¥	euro, cent, pound, yen strings	euro, cent, pound, yen
130, 132	single/ double curly low-9 quotes	‚ „	single/ doule straight quote	' "
133	horizontal ellipsis	…	a single dot	.
139, 155, 171, 187	single/ double left/ right-pointing angle quotation marks	‹ › « »	single/ double straight quote	' "
145, 146	left/ right single curly quotes	‘ ’	single straight quote	'
147, 148	left/ right double curly quotes	“ ”	double straight quote	"
150, 151	en, em dash	– —	dash	-
161, 191	inverted exclamation, question marks	¡ ¿	exclamation, questions marks	!, ?
180	acute accent	´	straight single quote	'
192-197, 224-229	Latin capital/ small A with grave, acute, circumflex, tilde, diaeresis and ring above	À, Á, Â, Ã, Ä, Å, à, á, â, ã, ä, å	Latin capital/ small A	A, a
198, 230	Latin capital AE	Æ, æ	Latin capital AE	Ae, ae
199, 231	Latin capital/ small C with cedilla	Ç ç	Latin capital/ small C	C, c
208, 240	Latin capital/ small Eth	Ð, ð	Latin capital/ small D	D, d
200-203, 232-235	Latin capital/ small E with grave, acute, circumflex, and diaeresis	È, É, Ê, Ë, è, é, ê, ë	Latin capital/ small E	E, e
131	Latin small f with hook	*ƒ*	Latin small f	f
204-207, 236-239	Latin capital/ small I with grave, acute, circumflex, and diaeresis	Ì, Í, Î, Ï, ì, í, î, ï	Latin capital/ small I	I, i
210-214, 242-246	Latin capital/ small O with grave, acute, circumflex, tilde and diaeresis	Ò, Ó, Ô, Õ, Ö, ò, ó, ô, õ, ö	Latin capital/ small O	O, o
140, 156	Latin capital/ small ligature OE	Œ, œ	Latin capital/ small OE	OE, oe
209, 241	Latin capital/ small N with tilde	Ñ, ñ	Latin capital/ small N	N, n
138, 154	Latin capital/ small S with caron	Š, š	Latin capital S	S, s
223	Latin small sharp s	ß	Latin small s	S
222, 254	Latin capital thorn	Þ, þ	Latin capital Th	Th
217-220, 249-252	Latin capital/ small U with grave, acute, circumflex, and diaeresis	Ù, Ú, Û, Ü, ù, ú, û, ü	Latin capital/ small U	U, u
159, 255, 221, 253	Latin capital letter/ small Y with diaeresis/ acute	Ÿ, ÿ, Ý, ý	Latin capital/ small Y	Y, y
142, 158	Latin capital Z with caron	Ž, ž	Latin capital Z	Z, z
134-137,		†, ‡, ˆ, ‰, •,		

149, 152, 153, 155, 160, 164, 166, 167, 168-170, 172, 174-179, 181, 186, 188-190, 215, 216, 247, 248	characters unlikely to appear in folktales	˜, ™, ›, , ¤,¦, §, ¨, ©, ª, ¬, ®, ¯,°,±, ²,³ , µ, ¶, ·, ¸, ¹, º, ¼, ½, ¾, ×, ø, ÷, ø	single white space
	backticks	``	straight double quote "

Table 1. Latin-1 supplement characters that were replaced with the ASCII compatible equivalents [116]

[116] http://www.alanwood.net/demos/ansi.html

10.2 *Appendix B: Part of Speech Tags*

Tag	Description	Tag	Description
CC	Coordinating conjunction	PRP$	Possessive pronoun
CD	Cardinal number	RB	Adverb
DT	Determiner	RBR	Adverb, comparative
EX	Existential there	RBS	Adverb, superlative
FW	Foreign word	RP	Particle
IN	Preposition or subordinating conjunction	SYM	Symbol
JJ	Adjective	TO	To
JJR	Adjective, comparative	UH	Interjection
JJS	Adjective, superlative	VB	Verb, base form
LS	List item marker	VBD	Verb, past tense
MD	Modal	VBG	Verb, gerund or present participle
NN	Noun, singular or mass	VBN	Verb, past participle
NNS	Noun, plural	VBP	Verb, non-3rd person singular present
NNP	Proper noun, singular	VBZ	Verb, 3rd person singular present
NNPS	Proper noun, plural	WDT	Wh-determiner
PDT	Predeterminer	WP	Wh-pronoun
POS	Possessive ending	WP$	Possessive wh-pronoun
PRP	Personal pronoun	WRB	Wh-adverb

Table 2. Alphabetical list of part-of-speech tags used in the Penn Treebank Project[117]

[117] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

10.3 *Appendix C: List of Folktale Books for the Project Corpus*

Source	Title	Author	First published	Link
Project Gutenberg	Australian			
	Australian Legendary Tales: Folklore of the Noongahburrahs as told to the Piccaninnies	K. Langloh Parker	1896	https://www.gutenberg.org/ebooks/3833
	German			
	Household Tales by Brothers Grimm	Grimm Jakob Ludwig Karl	1812	https://www.gutenberg.org/ebooks/5314
	Hawaiian			
	Hawaiian Folktales	Compiled by Thos Thrum	1907	https://www.gutenberg.org/ebooks/18450
	Indian			
	East Indian Fairy tales	Hartwell James	1906	https://www.gutenberg.org/ebooks/37708
	Folktales of Bengal	Lal Behari Day	1883	https://www.gutenberg.org/ebooks/38488
	Folklore of the Santal Parganas	Cecil	1909	https://www.gutenberg.org/ebooks/11938
	Hindu Tales from Sanskrit	N. Bell, S. Mitra	1919	https://www.gutenberg.org/ebooks/11310
	Indian Fairy Tales	Joseph Jacobs	1892	https://www.gutenberg.org/ebooks/7128
	Japanese			
	Japanese Fairy Tales	Yei Theodora Ozaki	1908	https://www.gutenberg.org/ebooks/4018
	Japanese Fairy Tales	Grace James	1910	https://www.gutenberg.org/ebooks/35853
	Jewish			
	Jewish Fairy Tales and Legends	Gertrude Landa	1919	https://www.gutenberg.org/ebooks/26711
	Russian			
	Old Peter's Russian Tales	Arthur Ransome	1916	https://www.gutenberg.org/ebooks/16981
	Russian Fairy Tales: A Choice Collection of Muskovite Folklore	W. R. S. Ralston	1887	https://www.gutenberg.org/ebooks/22373
	Russian Fairy Tales from the Skazki of Polevoi	R. Nisbet Bain	1901	https://www.gutenberg.org/ebooks/34705
	Russian Garland, Being Russian Folk Tales	Robert Steele	1921	https://www.gutenberg.org/ebooks/30109
	Spanish and Portuguese			
	Tales from the Lands of Nuts and Grapes	Charles Sellers	1888	https://www.gutenberg.org/ebooks/31481
	Portuguese			
	The Islands of Magic: Legends, Folk and Fairy Tales from the Azores	Elsie Eells	1922	https://www.gutenberg.org/ebooks/34431
	Dutch			
	The Flying Dutchman and Other Folktales from the Netherlands	Theo Meder	2008	N/A

Table 3. List of folktale Books to create the project corpus

10.4 *Appendix D: Feature Extraction*

Experiments	*Rapunzel* transformation after feature extraction
Initial document (after removing/ fixing punctuations, pet names and stop words) No. of food related terms/ No. of terms = 8/ 919	be man woman long vain wish child length woman hop God be to grant her desire people little window back their house splendid garden could be see be full beautiful flower herb It be however surround high wall one dare to go it it belong to enchantress great power be dread world One day woman be stand window look garden she saw bed be plant beautiful **rampion** it look fresh green she long it greatest desire to eat desire increase every day she know she could get it she quite pin away look pale miserable her husband be alarm ask aileth you dear wife ah she reply I cannot get **rampion** be garden behind our house to eat I shall die man love her think Sooner let thy wife die bring her **rampion** yourself let it cost you it twilight evening he clamber wall garden enchantress hastily clutch handful **rampion** take it to his wife she make herself salad it eat salad it much relish she however like it much much next day she long it three time much he be to rest her husband must descend garden gloom evening therefore he let himself he clamber wall he be terribly afraid he saw enchantress standing him you dare say she angry look to descend my garden steal my **rampion** like thief you shalt suffer it ah answer he let mercy take place justice I make my mind to it necessity My wife saw your **rampion** window felt longing it she would die she get to eat enchantress allow her anger to be soften say to him case be you sayest I allow you to take away you much **rampion** you I make one condition you must give me child thy wife bring world it shall be well treat I care it like mother man his terror consent to everything woman be bring to bed enchantress appear give child name Rapunzel take it away her Rapunzel grow beautiful child beneath sun she be twelve year old enchantress shut her tower lay forest neither stair door quite top be little window enchantress want to go she place herself beneath it cry Rapunzel Rapunzel Let thy hair to me Rapunzel magnificent long hair fine spin gold she hear voice enchantress she unfasten her braided tress wind them round one hook window hair fell twenty ell enchantress climb it year two it come to pass King s son ride forest go tower he hear song be charming he stand still listen be Rapunzel her solitude pass her time let her sweet voice resound King s son want to climb to her look door tower none be to be find He ride home singing deeply touch his heart every day he go forest listen to it he be thus stand behind tree he saw enchantress come he hear she cry Rapunzel Rapunzel Let thy hair Rapunzel let braid her hair enchantress climb to her be ladder one mount I try my fortune say he next day it begin to grow dark he go to tower cry Rapunzel Rapunzel Let thy hair Immediately hair fell King s son climb first Rapunzel be terribly frighten man her eye never yet behold come to her King s son begin to talk to her quite like friend tell her his heart be stir it let him rest he be force to see her Rapunzel lose her fear he ask her she would take him her husband she saw he be young handsome she think He love me old Dame Gothel she say yes lay her hand his She say I willingly go away you I know to get bring you skein silk every time you comest I weave ladder it be ready I descend you take me on thy horse They agree time he come to her every evening old woman come day enchantress remark nothing Rapunzel say to her tell me Dame Gothel it happen you be much heavier me to draw young King s son he be me moment ah you wicked child cry enchantress I hear you say I think I separate you world yet you deceive me her anger she clutch Rapunzel s beautiful tress wrap them twice round her left hand seize pair scissors right snip snap they be cut lovely braid lay on ground she be pitiless she take poor Rapunzel desert she to live great grief misery On day however she cast Rapunzel enchantress evening fasten braid hair she cut to hook window King s son come cry Rapunzel Rapunzel Let thy hair she let hair King s son ascend he find his

	dearest Rapunzel enchantress gaze him wicked venomous look Aha she cry mockingly you would fetch thy dearest beautiful bird sit longer singing nest cat get it scratch thy eye well Rapunzel be lose to you you never see her King s son be beside himself pain his despair he leap tower He escape his life thorn he fell pierce his eye he wander quite blind forest **eat** nothing root **berry** nothing lament weep loss his dearest wife Thus he roam misery year length come to desert Rapunzel twin to she give birth boy girl live wretchedness He hear voice it seem familiar to him he go towards it he approach Rapunzel know him fell on his neck weep Two her tear wet his eye they grow clear he could see them He lead her to his kingdom he be joyfully receive they live long time afterwards happy contented
Mini document (after picking all nouns and verbs) No. of food related terms/ No. of terms = 8/ 525	be man woman wish child length woman hop God be grant desire people window back house garden be see be flower herb be surround wall one dare go belong enchantress power be dread world day woman be stand window look garden saw bed be plant **rampion** look long desire eat desire increase day know get pin look husband be alarm ask aileth wife ah reply cannot get **rampion** be garden house **eat** die man love think let wife die bring **rampion** let cost twilight evening clamber wall garden enchantress clutch handful **rampion** take wife she make salad eat salad relish she like day long time be rest husband descend garden gloom evening let clamber wall be saw enchantress standing dare say look descend garden steal **rampion** thief suffer ah answer let mercy take place justice make mind necessity wife saw **rampion** window felt longing die get eat enchantress allow anger be soften say case be sayest allow take **rampion** make condition give child wife bring world be treat care mother man terror consent everything woman be bring bed enchantress appear give child name Rapunzel take Rapunzel grow child sun be year enchantress shut tower lay forest stair door top be window enchantress want go place cry Rapunzel Rapunzel Let hair Rapunzel hair fine spin gold hear voice enchantress unfasten tress wind round hook window hair fell ell enchantress climb year come pass King son ride forest go tower hear song be stand listen be Rapunzel solitude pass time let voice resound King son want climb look door tower none be be find ride home singing touch heart day go forest listen be stand tree saw enchantress come hear cry Rapunzel Rapunzel Let hair Rapunzel let braid hair enchantress climb be ladder one mount try fortune say day begin grow go tower cry Rapunzel Rapunzel Let hair hair fell King son climb Rapunzel be frighten man eye behold come King son begin talk friend tell heart be stir let rest be force see Rapunzel lose fear ask take husband saw be think love Dame Gothel say lay hand say go know get bring skein silk time comest weave ladder be descend take on horse agree time come evening woman come day enchantress remark nothing Rapunzel say tell Dame Gothel happen be draw King son be moment ah child cry enchantress hear say think separate world deceive anger clutch Rapunzel tress wrap round hand seize pair scissors right snip snap be cut braid lay on ground be take Rapunzel desert live grief misery day cast Rapunzel enchantress evening fasten braid hair cut hook window King son come cry Rapunzel Rapunzel Let hair let hair King son ascend find Rapunzel enchantress gaze look Aha cry fetch dearest bird sit singing nest cat get scratch eye Rapunzel be lose see King son be pain despair leap tower escape life thorn fell pierce eye wander forest eat nothing root **berry** nothing lament weep loss wife roam year length come desert Rapunzel twin give birth boy girl live wretchedness hear voice seem go approach Rapunzel know fell on neck weep tear wet eye grow see lead kingdom be receive live time
Micro (only food nouns, verb). No.f.t/ No.terms 9/9	rampion rampion rampion rampion rampion rampion rampion eat berry

Table 4. Feature extraction through subsequent experiments

10.5 *Appendix E: Basic Abbreviations*

POS = Part of Speech

VE = Virtual Example

SVM = Support Vector Machine

NB = Naive Bayes

ATU = Aarne/ Thompson/ Uther

TMI = Thompson motif-index

NLTK = Natural Language Toolkit

ASCII = American Standard Code for Information Interchange

ML = Machine Learning

RB = Rule-based

F = Fruitarian

N = Neutral

O = Omnivorous

V = Vegan

Vg = Vegetarian

Table 5. The most frequently used abbreviations