Human-Centric Justification of Machine Learning Predictions

Or Biran*
n-Join
or@n-join.com

Kathleen McKeown
Columbia University
kathy@cs.columbia.edu

Abstract

Human decision makers in many domains can make use of predictions made by machine learning models in their decision making process, but the usability of these predictions is limited if the human is unable to justify his or her trust in the prediction. We propose a novel approach to producing justifications that is geared towards users without machine learning expertise, focusing on domain knowledge and on human reasoning, and utilizing natural language generation. Through a task-based experiment, we show that our approach significantly helps humans to correctly decide whether or not predictions are accurate, and significantly increases their satisfaction with the justification.

1 Introduction

Machine learning systems are increasingly used to assist humans in decision making, producing predictions or recommendations which are then considered by a human decision-maker. It is important that the prediction can be *justified*: the user should understand why the system produced its prediction, and make sure she agrees, before making a decision.

For the rule-based expert systems that were prevalent at the end of the 20th century, it is often enough to *explain* how the system reached its conclusion. The human user will be able to understand the set of rules governing the prediction, and given the proper information about the particular situation leading to a specific prediction (the relevant states of the data and the chain of rules that led to the final decision), will be able to make up his mind about the prediction's validity. This is called the "white box" model, in contrast to the "black box" model where explanation is not given.

Recently, machine learning (ML) techniques have all but replaced rule-based methods, resulting in increased accuracy and an ability to handle more complex problems. In contrast to rule-based systems, justifying the predictions of ML is not a straightforward task: it is no longer the case that *explaining* how a prediction was reached automatically *justifies* it to the user. Due to the complex, quantitative and unintuitive nature of most models, it is unreasonable to expect that users who are not ML experts, even if they are experts in the domain of the prediction, will understand how the model works, regardless of how transparently it is presented. In other words, the white box model is no longer useful for most users.

*Work done while at Columbia University

A black box with no justification at all, however, is even worse. In this paper, we propose a model where Natural Language Generation (NLG) is used to produce simple, short, qualitative and intuitive justifications for ML predictions, relying on the domain knowledge embodied in the features.

Work on *prediction interpretation* [Robnik-Sikonja and Kononenko, 2008; Ribeiro *et al.*, 2016], which is concerned with creating interpretable (e.g. linear) models that approximate an uninterpretable model's behavior for a particular prediction is complementary to ours. In fact, our approach expects an interpretable model for the prediction we are justifying. Note that "interpretable" in this context means interpretable by a ML expert or a statistician: a model that has a low enough complexity to be explored in finite time by an expert, for example through visualization. What we focus on in this paper is creating a general human-centered narrative from the quantitative details of the model - a narrative not about the model, but about the evidence that led to the prediction. We also present the first work, to our knowledge, that takes into account *missing evidence*, which is an important part of human decision making but often ignored in previous work on explanation. We show, through an experiment with human subjects, that our approach helps users accurately judge whether or not a prediction is correct, as well as increasing their satisfaction with the explanation. We have made this work available to the community as an open-source library.

2 Related Work

Related work for producing justifications (or more commonly *explanations*, which are presumed to be justifying) come from multiple fields. Historically, explanations first appeared in the context of rule-based expert systems [Shortliffe and Buchanan, 1975; Swartout, 1983; Barzilay *et al.*, 1998], and were mostly treated as a systems design task (i.e., designing a system capable of producing explanations and drill-down into its decisions). In some fields - especially the medical - probabilistic decision-making systems are still called expert systems and their explanation is treated as a continuation of that line of research [Lacave and Díez, 2002; Yap *et al.*, 2008; Helldin and Riveiro, 2009].

In the 2000's justification has also been of particular importance in the field of recommender systems, where a ML or other probabilistic system needs to justify its recommendation to many users. [Herlocker *et al.*, 2000] measured user satisfaction with a variety of justification types for a movie recommendation system. They found that the most satisfying were simple and conclusive methods, such as stating the neighbors' ratings or focusing on a single strong feature like a favorite actor. Surprisingly, justifications incorporating ML concepts such as model confidence and complex justifications such as a full neighbor graph scored *lower than the baseline black box*. Other studies have also shown that users are overwhelmingly more satisfied with systems that contain some form of justification [Sinha and Swearingen, 2002] and that feature-based justifications are superior to those that rely on user history, neighors or model details [Bilgic and Mooney, 2005; Symeonidis *et al.*, 2009; Papadimitriou *et al.*, 2012].

In the machine learning literature, work on explanation has often focused on producing visualizations of the prediction in order to assist ML experts in evaluating the correctness of the model. One very common visualization technique is *nomograms*. It was first applied to logistic regression models by [Lubsen *et al.*, 1978], and later to Naive Bayes [Možina *et al.*, 2004], SVM [Jakulin *et al.*, 2005] and other models. [Szafron *et al.*, 2003] proposed a more detailed visualization-based framework for Naive Bayes.

Other work has focused on interpreting the predictions of specific complex models, often by proposing to isolate the contributions of individual features to the prediction. Such proposals were made for Bayesian networks [Suermondt, 1992], multi-layer Perceptrons [Feraud and Clerot, 2002], SVMs [Carrizosa *et al.*, 2006] and other models.

In addition to model-specific methods, there have been suggestions for model-agnostic frameworks. [Robnik-Sikonja and Kononenko, 2008] and [Kononenko *et al.*, 2013] measure the effect of a feature on an unknown classifier's prediction by checking what the prediction would have been if that feature value was absent and comparing the two using various distance measures. The effects are displayed visually to show the main contributors towards a prediction or to compare the feature's effect in various models. [Baehrens *et al.*, 2010] describe an alternative approach using *explanation vectors* (class probability gradients) which highlight the effect of the most important features. [Ribeiro *et al.*, 2016] introduced a method which learns an interpretable (linear) model locally around the prediction, approximating how the global model behaves in that region.

Further removed yet relevant lines of research include computational argumentation [Rahwan and Simari, 2009]; explanation of statistical information in forensic science [Vlek *et al.*, 2016; Timmer *et al.*, 2017]; and explanation in context-aware applications [Tullio *et al.*, 2007; Lim and Dey, 2010].

3 Justification Narratives

While it is not reasonable to expect non-experts to understand the details of how a prediction was made by a ML model, it is still important that they understand the variables affecting the prediction enough to satisfy the question of justification. It has been shown that evidence-based causal models of justification are more satisfactory to users than white box models and that replacing numeric values with qualifying linguistic expressions (high, strong, etc) also enhances satisfaction [Druzdzel, 1996; Herlocker *et al.*, 2000; Lacave and Díez, 2002]. The features used in ML models often correspond to real-world evidence that non-experts understand well (and when they do not, such real-world evidence can sometimes be approximated, e.g. as shown by [Ribeiro *et al.*, 2016]). A justification for a prediction can rely on these variables, their importance to the model, their effect on the prediction, and their interactions.

A robust method of automatically generating prediction justification, then, should focus on selecting the most important pieces of evidence for the prediction and on analyzing and presenting their roles in the prediction. The selected evidence should be presented to the user in a qualitative way that is invariant across different models and readily understandable by a non-expert.

This section describes a framework for producing *justification narratives* from ML predictions. Each narrative is composed of a subset of the relevant features, where each feature has a discrete *role*. The framework enables us to more easily decide what an appropriate feature subset looks like and to present it in a way that is more qualitative than quantitative. This plays a role somewhat similar to the Signal Analysis and Data Interpretation modules described in [Reiter, 2007] for data-to-text generation systems.

3.1 Narrative Roles

The first step in producing a narrative is determining the role of each feature. One important concept that is well-explored in the literature is a feature's *effect* on the prediction - its contribution towards or against the predicted class [Robnik-Sikonja and Kononenko, 2008; Carrizosa *et al.*, 2006; Yap *et al.*, 2008]. In addition, we are interested in a feature's *importance* in the model - the expected effect of the feature for a prediction of the particular class, regardless of its actual value in the current prediction.

In this paper, we focus on linear classifiers which utilize *linear discriminant functions*. This family contains many commonly-used ML classifiers, including Logistic Regression, Perceptrons and Linear SVMs, and log-linear models such as Naive Bayes can also be formulated in a way that fits this framework. For more complex models (e.g. Kernel SVMs or Neural Nets), a linear model that approximates the local behavior can be found using the a method like the one proposed by [Ribeiro *et al.*, 2016]. Alternatively, a feature's effect on a prediction (and with many predictions, its expected effect) can be approximated in other ways [Kononenko *et al.*, 2013].

A linear discriminant function for data instance x in the general multi-class linear classifier is

$$f(y,x) = \sum_i \theta_{yi} x_i$$

Where each x_i is a feature value and the coefficients θ_{yi} have been learned from the training data for each class y us-

Importance \ Effect	High positive	Low	High negative
High positive	Normal evidence	Missing evidence	Contrarian counter-evidence
Low	Exceptional evidence	Negligible	Exceptional counter-evidence
High negative	Contrarian evidence	Missing counter-evidence	Normal counter-evidence

Table 1: Narrative roles assignment for the range of feature effect and importance

ing some learning algorithm, and may include an intercept.[1] The classifier predicts the class of the instance as the one that maximizes the predictor function, possibly through a monotonic non-linear distortion function φ:

$$\begin{aligned}\hat{y} &= \underset{y}{\operatorname{argmax}}\, \varphi\left(f(y,x)\right)\\ &= \underset{y}{\operatorname{argmax}}\, f(y,x)\end{aligned}$$

The linear *effect* of a feature i towards or against predicting class y for a data instance is the product of the feature's coefficient and its value in the instance:

$$\text{Effect}_{yi} = \theta_{yi} x_i$$

While its *importance*, the expected effect for predictions of the class, can be estimated using the mean feature value for the class (X^y is the set of all instances in the training set with class y):

$$\text{Importance}_{yi} = \theta_{yi} \frac{\sum_{x \in X^y} x_i}{|X^y|}$$

Intersecting these two concepts, we assign a *narrative role* for each feature. Narrative roles are assigned based on the sign and magnitude of the importance and effect of a feature for the predicted class. They represent semantically clear concepts that non-experts readily understand, and are rooted in the true details of the prediction. Table 1 shows the roles for all possible combinations.

Table 1 assumes that we have discretized the importance and effect values of the features into *high positive*, *high negative* and *low*. We discretize the values as follows.

To discretize importance, we first separate the features with positive and negative importance in the model. For each of these groups, we normalize the (absolute) importance values of the features, and find the smallest subset H such that

$$\sum_{i \in H} \text{Importance}_{yi} > \tau$$

where τ is a tunable parameter that we set to 0.75 in our experiments. We then set the high/low threshold halfway between the lowest importance of any feature in H and the highest importance of a feature not in H. In other words, the smallest set of features that together have over 75% of the total importance of the features (on the positive side, and separately on the negative side) have high importance, while all others have a low importance. The same threshold is used to discretize the effect of features for each instance. If the feature importance values have a non-uniform distribution, this strategy optimizes succinctness and informativeness.

The initial content selection for a justification is to exclude *negligible* features. Features that have one of the other eight roles represent the most important evidence which a user will need to see in order to make up her mind about the prediction: they either had a direct effect on the prediction, or else they were expected to but did not.

Narrative Role Descriptions

Narrative roles can be broadly divided into three groups: evidence, missing evidence, and counter-evidence roles.

Features having evidence roles are those that had a positive contribution toward making the prediction. **Normal evidence** is evidence that is expected to be present in many instances predicted to be in the class: with high importance, high effect is not surprising. **Exceptional evidence** is evidence that is not usually expected to be present. With low importance, high effect means that the feature value is surprisingly high. The user may decide that this makes the prediction dubious (especially if the result would be different if this feature were within the normal range), or on the contrary - perhaps this feature represents rare but solid causal evidence. **Contrarian evidence** is strongly unexpected, since the effect has the opposite sign than expected. Note that contrarian evidence (and contrarian counter-evidence) is only possible for features that may have negative values, and may not appear in many real-world applications.

A feature that has **Missing evidence** as a role is an important feature that was expected to contribute positively, but was weak for the particular instance. We want to include it in the narrative because it means that the prediction was uncharacteristically made without the typically important contribution of that feature (depending on the prediction domain, this may tell the human that this prediction is likely to be wrong. It may also signal the opposite - that the other evidence for the prediction is exceptionally sound). Similarly, **missing counter-evidence** is given to features that were expected to contribute negatively but did not.

Finally, there are counter-evidence roles. Features having these roles contributed against the prediction, although they were not strong enough to overturn it. They are important because the human may think one or more of these are reason to mistrust the prediction, even though the evidence features had a higher effect in the model. **Normal counter-evidence** is expected: it is normal for this feature to have a high negative effect, even if the positive effects from other features ultimately overpower it. **Exceptional counter-evidence** is

[1]This is a generalization of the binary classifier case, where there is only one coefficient vector θ for the positive class, and the negative class implicitly has the negation of the positive class coefficients

unusual. Finally, **contrarian counter-evidence** means that a feature we normally expect to contribute positively contributes negatively instead.

4 Generation

Given a model definition, a data instance and a prediction, we generate a text comprised of two types of messages. *Core messages* are messages about the roles of key features, as determined by the justification narrative created for the prediction as detailed in the previous section. These are realized using hand-crafted sets of templates for the eight possible role types. For example, "[feature] exhibits typical evidence in support of the prediction" is one of the templates for the *normal evidence* role. Each role descriptions has 5 associated templates which are paraphrases of one another.

In addition, the justification contains *secondary messages* which define and describe the importance (not in terms of weight in the model or the prediction, but in a real-world sense) of key features. These aim to explain to the reader, in plain language, why the feature might be important. Secondary messages are definitional sentences that were extracted ahead of time from Wikipedia, in a process that only takes place once for each model whose predictions are to be justified: Wikipedia is scanned for sentences that contain a feature name as a subject; if the sentence is below a length threshold and does not contain more than one other proper noun (to avoid sentences that are complex and context-heavy), it is extracted and kept as a secondary message about the feature, which can be added to justifications where the feature appears. Because features are not always given names that correspond to real-world concepts, secondary messages in general are not available for all features.

Content selection, therefore, is done in two parts: first, core messages are created according to the narrative; then, a subset of possible secondary messages for the participating features is selected using [Barzilay and Lapata, 2005]'s energy minimization framework approach, by minimizing

$$\sum_{m \in S} ind(m) + \sum_{\substack{m_i \in S \\ m_j \in N}} link(m_i, m_j)$$

where S is the subset of selected secondary messages and N is the subset of unselected messages;

$$ind(m) = -\frac{1}{\frac{\sum_{c \in C} J(c,m)}{|C|}}$$

where C is the set of core messages and $J(c, m)$ is the Jaccard Coefficient between the member entities of messages c and m; and $link(m_i, m_j)$ is the Jaccard Coefficient between the *types* of the member entities of messages m_i and m_j.

Minimizing this function means that we tend towards selecting more secondary messages that contain the exact entities contained by many core messages; more secondary messages that share entity types with other secondary messages (so that a justification tends to focus on one or a few *themes*); and few secondary messages overall (since the individual scores are always negative).

To order the selected messages (both core and secondary), we use the discourse model of [Biran and McKeown, 2015], which gives transitional probabilities between discourse relations that potentially hold between messages (they use the four top-level relations of the Penn Discourse TreeBank [Prasad *et al.*, 2008], of which we are only interested in two: *expansion* and *comparison*). To use this model, we define which discourse relations are allowed to occur between messages as follows:

1. Possible *comparison* relation for two core messages if one describes an evidence role (including missing counter-evidence) and the other describes a counter-evidence role (including missing evidence)
2. Possible *expansion* relation for two core messages if both describe an evidence role, or if both describe a counter-evidence role
3. Possible *expansion* for any two messages which describe the same feature (most relevant to secondary messages)
4. Possibly no relation for any two messages

The first message to be generated is a special one that states the prediction result. From there, we let the model decide stochastically what follows, but we augment [Biran and McKeown, 2015]'s original model by weighting the probabilities by $J(c, m)$ as defined above, to encourage messages about the same entities to appear together. Note that the model gives us both the ordering and the discourse relations (if any) between messages.

To realize the ordered messages as text, we use a cross-sentence bigram language model extracted from Wikipedia, and choose the template to use for each message stochastically (given the template chosen for the previous message).

For messages that have discourse relations between them, we generate a discourse connective from a short list of 7 templates for each relation type (e.g., "m_i. In addition, m_j" is one template for the *expansion* relation). We choose the empty template (no connective) with 50% probability, and the others initially have an equal chance of being chosen. Each time a connective is used, its probability of being chosen again is halved.

5 Evaluation

To see if our generated justifications really help the end-user in correctly deciding whether or not the prediction is correct, we devised a task based evaluation. Our subject domain in this evaluation is a *stock price prediction* classifier (Logistic Regression) which predicts whether the price of a stock is going to rise or fall in the following 10 business days (that is, whether the close price after 10 business days will be higher or lower than the current close price). The classifier uses the 23 features shown in Table 2 and is trained on two years of daily pricing data for the S&P500 companies, available on Yahoo! Finance. The accuracy of the classifier (for the same S&P500 companies, for predictions made on the day following the training period) is 58.5% (F1 is 68.7). In our evaluation, we present a prediction about the price of an anonymized stock to a human, along with a justification or a baseline and ask whether she would buy or avoid this stock. The task is

Technical Analysis Signals	2 day high / current 5 day high / current 10 day high / current 30 day high / current	2 day momentum 5 day momentum 10 day momentum 30 day momentum
Prior Returns	2 day returns 5 day returns 10 day returns	neutral 2day returns neutral 5day returns neut. 10day returns

Valuation Multiples	price/earnings ratio
	price/book ratio
	ev/ebitda ratio
	return on equity
	debt to equity ratio
		price/sales ratio
		dividend yield
		return on assets
		return on revenue

Table 2: Features used by the stock price prediction classifier.

to make as much money as possible betting on the stocks; we hypothesize that humans who saw our generated justification will make more accurate decisions, and therefore make more money, than those who saw the baseline.

For each prediction in our evaluation, we create four justification versions. In the first, we show only the prediction and no justification. In the second, we show the prediction with a graphic representation of the effect of key features. In the third, we give the prediction along with a textual justification, and in the fourth, we provide both the graphic representation and the textual justification. In that way, we evaluate the relative importance of the textual justification we propose in this paper and a more traditional graphic visualization (in fact, the graphic visualization is very similar to the one used by [Ribeiro *et al.*, 2016]). An example of the full justification, with the textual and graphic components, is shown in Figure 1. The text-only and graphic-only versions look the same, except they consist of only one of the parts.

We used predictions made by the classifier for 487 members of the S&P500 (pricing data was missing for the other 13) on July 7th, 2016 predicting the difference in price on July 21st, 2016. We then conducted a crowd-sourced experiment on CrowdFlower where annotators (taskers) were shown the prediction for one of the stocks (they were anonymized to avoid bias from any real-world knowledge the annotators may have had) along with one of the four justification options: prediction only; graphic only; text only; or text + graphic. The annotators, who had no special experience with financial investment or ML, were asked three questions: whether they would buy or avoid the stock, given the prediction and justification they saw; whether or not the information provided helped them in making their choice; and to what extent (on a scale of 1-5) they were satisfied with the information they were presented. Overall, we had 33 annotators answer $1,948$ questions. To keep it interesting and encourage the annotators to behave like investors, we offered (relatively) large bonuses to the two annotators who made the most virtual money.

The results are shown in Table 3. We show the average accuracy, precision, recall, F1 and financial returns achieved by the annotators for each justification category (these were measured based on the agreement of the annotator's choice with the true returns of the stock, regardless of the classifier's prediction), as well as the ratio of annotators' agreement with the prediction for each justification category. In addition, we show the ratio of annotators who said the information helped them in making the decision, and the average satisfaction rating for each category.

	None	Graphic	Text	Both
Accuracy	23.85%	28.67%	29.1%	32.17%
Precision	56.57	63.16	64.57	68.09
Recall	15.51	23.27	22.71	26.59
F1	24.35	34.01	33.61	38.25
Returns	1.71%	2.1%	2.79%	2.42%
Agreement	85.22%	81.93%	83.98%	81.31%
Help	45.79%	78.44%	75.36%	84.39%
Satisfaction	2.54	3.17	3.21	3.38

Table 3: Results of the task-based evaluation. Precision, Recall and F1 are measured with respect to the positive class.

For the accuracy metrics, all differences between categories are statistically significant with the exception of the difference between *grapical only* and *text only*, and the difference between *text + graphic* and *text only*. For "help" and satisfaction, all differences are statistically significant except for the difference between *graphic only* and *text only*.

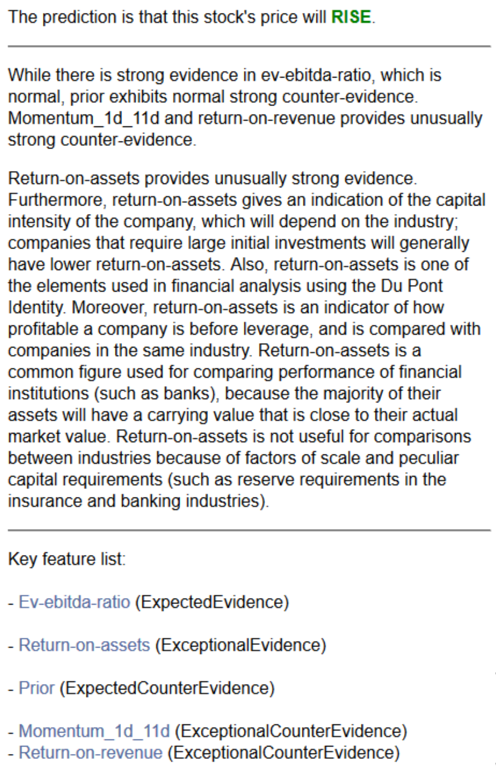
5.1 Discussion

The first thing to notice in Table 3 is that having any kind of justification significantly increases all accuracy measures as well as the financial returns achieved by the annotators. It also reduces the agreement with the prediction of the classifier, which suggests that the justification is doing what it is intended to: give annotators enough information to decide (in some cases) that the classifier is wrong.

It is interesting to see how low the annotators' recall is with respect to precision. This suggests that the annotators behaved like fearful investors: they chose to "buy" a stock more rarely. The classifier itself predicted that stocks will rise and fall approximately equally for the test period (in fact, with a slight advantage to rising) - this, combined with the relatively high agreement annotators had with the prediction, suggests that most deviations from the classifier's prediction were to *not* buy a stock even when it predicted it will rise, presumably because the justification was not convincing enough.

Looking at the three types of justification (graphic, textual, and both), it is clear that having a combined textual and graphic justification achieves the best results on all accuracy measures. While the returns achieved with text-only are higher than those achieved by the combined justification, keep in mind that the returns are heavily affected by the *amount* by which each stock increases or decreases, which is not modeled in any way by the classifier or the justification, and so are more volatile than the accuracy metrics.

We were not able to show statistical significance for any of the differences between the graphic-only and the textual-only variations: either the two are complementary, or different annotators have different preferences. It is clear, however, that having *both* is significantly better than having just the graphic justification, on all counts - accuracy metrics (including actual financial returns), helpfulness, and satisfaction - which shows the value of using justification narratives, actualized via NLG, for justifying predictions.



The prediction is that this stock's price will **RISE**.

While there is strong evidence in ev-ebitda-ratio, which is normal, prior exhibits normal strong counter-evidence. Momentum_1d_11d and return-on-revenue provides unusually strong counter-evidence.

Return-on-assets provides unusually strong evidence. Furthermore, return-on-assets gives an indication of the capital intensity of the company, which will depend on the industry; companies that require large initial investments will generally have lower return-on-assets. Also, return-on-assets is one of the elements used in financial analysis using the Du Pont Identity. Moreover, return-on-assets is an indicator of how profitable a company is before leverage, and is compared with companies in the same industry. Return-on-assets is a common figure used for comparing performance of financial institutions (such as banks), because the majority of their assets will have a carrying value that is close to their actual market value. Return-on-assets is not useful for comparisons between industries because of factors of scale and peculiar capital requirements (such as reserve requirements in the insurance and banking industries).

Key feature list:

- Ev-ebitda-ratio (ExpectedEvidence)
- Return-on-assets (ExceptionalEvidence)
- Prior (ExpectedCounterEvidence)
- Momentum_1d_11d (ExceptionalCounterEvidence)
- Return-on-revenue (ExceptionalCounterEvidence)

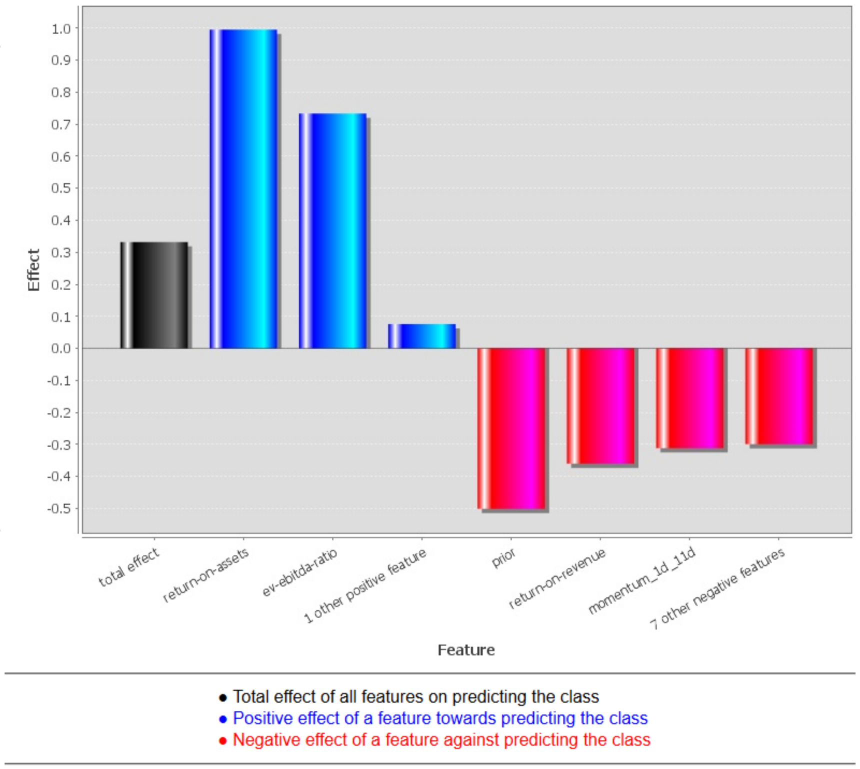


Figure 1: Example of a full justification produced for a prediction in the evaluation.

6 PreJu

PreJu is a Java package that generates justifications for classifier predictions using the method described in this paper. We have made it publicly available[2] for researchers who are interested in justifying their classifiers' predictions.

PreJu works in one of two ways: as a stand-alone configurable tool, it accepts input in the form of XML providing the effect and importance scores of the features, leaving the implementation details to the user, and allows the configuration of key feature selection, output types and other parameters via XML. As an API, it provides simple interfaces for producing justifications programmatically and currently contains implementations for Weka's Logistic Regression and SMO classifiers as integrated input sources.

7 Conclusion and Future Work

We tackled the relatively unexplored task of ML prediction justification via NLG. We proposed a novel human-centric method of determining the most important core information (and consequently, core messages) of the prediction justification task using *narrative roles*. We then applied this approach to a stock price prediction classifier and conducted a human study which showed our generated justification significantly enhances users' ability to determine whether or not the classifier prediction is accurate, as well as their satisfaction and inclination to say the justification is helpful. These effects hold when comparing a text-only justification to no justification, and also when comparing a text+graphic justification to a graphic-only justification, suggesting that the textual part is key to all three evaluation metrics. We made our justification method publicly available as a Java package.

While the work in this paper provides a solid proof of concept, there are clear paths to improving our method in future work. One phenomenon we neglect in our handling of feature contributions is the case of highly-correlated features: in such a scenario, we may miss some very important evidence because it has been separated into multiple features, all of which have relatively low weights in the classifier. A solution to this would be to use a feature aggregation method which combines highly correlated features into groups. A more optimal (from the classifier's point of view) solution, though more problematic from an interpretability point of view, would be a transformation such as Principal Components Analysis.

[2] `http://www.cs.columbia.edu/~orb/preju/`

References

[Baehrens *et al.*, 2010] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *JMLR*, 11, August 2010.

[Barzilay and Lapata, 2005] Regina Barzilay and Mirella Lapata. Collective content selection for concept-to-text generation. In *HLT/EMNLP*, Vancouver, 2005.

[Barzilay *et al.*, 1998] Regina Barzilay, Daryl Mccullough, Owen Rambow, Jonathan DeCristofaro, Tanya Korelsky, and Benoit Lavoie. A new approach to expert system explanations. In *International Workshop on NLG*, 1998.

[Bilgic and Mooney, 2005] Mustafa Bilgic and Raymond J. Mooney. Explaining recommendations: Satisfaction vs. promotion. In *Workshop on the Next Stage of Recommender Systems Research*, San Diego, CA, 2005.

[Biran and McKeown, 2015] Or Biran and Kathleen McKeown. Discourse planning with an n-gram model of relations. In *Empirical Methods in Natural Language Processing (EMNLP)*, Lisbon, Portugal, 2015.

[Carrizosa *et al.*, 2006] Emilio Carrizosa, Belen Martin, and Dolores Morales. A column generation approach for support vector machines. Technical report, 2006.

[Druzdzel, 1996] Marek J Druzdzel. Qualitiative verbal explanations in bayesian belief networks. *AISB QUARTERLY*, pages 43–54, 1996.

[Feraud and Clerot, 2002] Raphael Feraud and Fabrice Clerot. A methodology to explain neural network classification. *Neural Networks*, 15(2):237 – 246, 2002.

[Helldin and Riveiro, 2009] Tove Helldin and Maria Riveiro. Explanation methods for bayesian networks: review and application to a maritime scenario. In *Skövde Workshop on Information Fusion Topics*. University of Skövde, 2009.

[Herlocker *et al.*, 2000] Jonathan Herlocker, Joseph Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Computer Supported Cooperative Work (CSCW)*, 2000.

[Jakulin *et al.*, 2005] Aleks Jakulin, Martin Možina, Janez Demšar, Ivan Bratko, and Blaž Zupan. Nomograms for visualizing support vector machines. In *Knowledge Discovery in Data Mining (KDD)*, 2005.

[Kononenko *et al.*, 2013] Igor Kononenko, Erik Strumbelj, Zoran Bosnic, Darko Pevec, Matjaz Kukar, and Marko Robnik-Sikonja. Explanation and reliability of individual predictions. *Informatica (Slovenia)*, 37(1):41–48, 2013.

[Lacave and Díez, 2002] C. Lacave and F. J. Díez. A review of explanation methods for Bayesian networks. *Knowledge Engineering Review*, 17:107–127, 2002.

[Lim and Dey, 2010] Brian Lim and Anind Dey. Toolkit to support intelligibility in context-aware applications. In *Ubiquitous Computing (Ubicomp)*, 2010.

[Lubsen *et al.*, 1978] J. Lubsen, J. Pool, and E. van der Does. A practical device for the application of a diagnostic or prognostic function. *Methods of information in medicine*, 17(2):127–129, April 1978.

[Možina *et al.*, 2004] M. Možina, J. Demšar, M. Kattan, and B. Zupan. Nomograms for visualization of naive bayesian classifier. In *Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2004.

[Papadimitriou *et al.*, 2012] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos. A generalized taxonomy of explanations styles for traditional and social recommender systems. *Data Min. Knowl. Discov.*, 24(3):555–583, 2012.

[Prasad *et al.*, 2008] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. The penn discourse treebank 2.0. In *Proceedings of LREC*, 2008.

[Rahwan and Simari, 2009] Iyad Rahwan and Guillermo R. Simari. *Argumentation in Artificial Intelligence*. Springer Publishing Company, Incorporated, 1st edition, 2009.

[Reiter, 2007] Ehud Reiter. An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, ENLG '07, 2007.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Knowledge Discovery and Data Mining (KDD)*, 2016.

[Robnik-Sikonja and Kononenko, 2008] M. Robnik-Sikonja and I. Kononenko. Explaining classifications for individual instances. *Knowledge and Data Engineering, IEEE Transactions on*, 20(5):589–600, May 2008.

[Shortliffe and Buchanan, 1975] Edward H Shortliffe and Bruce G Buchanan. A model of inexact reasoning in medicine. *Mathematical biosciences*, 23(3), 1975.

[Sinha and Swearingen, 2002] Rashmi Sinha and Kirsten Swearingen. The role of transparency in recommender systems. In *CHI EA*, 2002.

[Suermondt, 1992] Henri Jacques Suermondt. *Explanation in Bayesian Belief Networks*. PhD thesis, Stanford, CA, USA, 1992. UMI Order No. GAX92-21673.

[Swartout, 1983] William R. Swartout. Xplain: A system for creating and explaining expert consulting programs. *Artif. Intell.*, 21(3), September 1983.

[Symeonidis *et al.*, 2009] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Moviexplain: A recommender system with explanations. In *RecSys*, 2009.

[Szafron *et al.*, 2003] Duane Szafron, Russell Greiner, Paul Lu, David Wishart, Cam Macdonell, John Anvik, Brett Poulin, Zhiyong Lu, and Roman Eisner. Explaining naive bayes classifications. Technical report, 2003.

[Timmer *et al.*, 2017] S. Timmer, J. Meyer, H. Prakken, S. Renooij, and B. Verheij. A two-phase method for extracting explanatory arguments from bayesian networks. *Int. J. Approx. Reasoning*, 80(C):475–494, January 2017.

[Tullio *et al.*, 2007] Joe Tullio, Anind Dey, Jason Chalecki, and James Fogarty. How it works: A field study of non-technical users interacting with an intelligent system. In *SIGCHI Human Factors in Computing Systems*, 2007.

[Vlek *et al.*, 2016] Charlotte S. Vlek, Henry Prakken, Silja Renooij, and Bart Verheij. A method for explaining bayesian networks for legal evidence with scenarios. *Artificial Intelligence and Law*, 24(3):285–324, 2016.

[Yap *et al.*, 2008] Ghim-Eng Yap, Ah-Hwee Tan, and Hwee-Hwa Pang. Explaining inferences in bayesian networks. *Applied Intelligence*, 29(3):263–278, 2008.