Peeking inside FFORMS: Feature-based FORecast Model Selection

Abstract

This study investigates the relationship between features of time series and forecast-model selection using FFORMS framework. It is becoming more of a challenge to not only build state-of-the-art predictive models, but also gain an understanding of what's really going on in the data. We explore the impact of features in three levels, globally, subset of observations and individual level. In this paper we use model-agnostic appraches to explore the impact of time series features towards forecast-model selection. Graphical representations are used to visualize both main and interaction effects.

Keywords: FFORMS, machine learning, interpretability, partia-dependece, time series

1 Introduction

The time series forecasting field has been evolving for a long time and has introduced a wide variety of forecasting methods. However, for a given time series the selection of an appropriate forecasting-method among many possibilities is not straight forward. This selection is one of the most difficult tasks as each method perform best for some but not all tasks. The features of time series are considered to be an important factor in identifying suitable forecasting models[ref, Meade, MakrikadisM3].[This goes back to 1972]. However, a description of relationship between the features and the performance of algorithms is rarely discussed in the field of forecasting.

There have been several recent studies on use of machine learning algorithms to automate the forecast model selection based on the features computed from the time series. Meta-learning approach provides a systematic guidance on model selection based on knowledge acquire from historical data set, in our case historical collection of time series. The key idea behind these framework is, forecast-model selection is posed as a supervised learning task. Each time series in the meta-data set is represented as a vector of features and labeled according to the "best" forecasting method(i.e. lowest MASE, etc.). Then a meta-learner is trained to identify suitable forecasting models. With the era of big data, such an automated model selection process is necessary because the cost of invoking all possible forecasting method is prohibitive. However,

these work suffer from the limitation of providing meaningful interpretations that can enhance understanding of relations between features and model outcomes. To best of our knowledge, very limited efforts have been taken to understand how the models are making its decisions and what is really happening inside these complex model structures. This results in less transparency of the model which lead to the questions of

1. How features are related to the property being modeled?
2. How features interact with each other to identify the suitable forecasting method?
3. Why one forecasting method is preferable over other?
4. How different features contribute to the model prediction or how they affect the model performance?
5. Why certain features were responsible in driving certain decisions?
6. Why is your model less accurate in some areas of the instance space?
7. Which features contribute the most to classify a specific instance?

On the other hand, aside from the goal of developing automated forecast-model selection framework few researchers have made an attempt to provide a description of relationship between the features and the performance of algorithms. One of the first attempts to identify the relationship between features and forecast-model performance was presented in xx. However, these studies are limited by the scale of problem instances used, diversity of forecasting-methods used, quality of features considered, and modelling approached used to identify the relationship between features and forecast model performance. Most of these studies are typically restricted to simple statistical techniques such as. . . [do not capture the complex interaction effects]

To explore these points further, this paper makes a first step towards providing a comprehensive explanation of the relationship between time series features and forecast-model selection using machine learning interpretability techniques. This paper builds on the method from our previous work ref, in which we introduced the FFORMS (Feature-based FORecast Model Selection) framework. The random forest algorithm is used to model the relationship between features and "best" performing forecast-model. [Large time series collection is used to train the model] We use 30 features to capture morphology of time series. Even though the prediction accuracy of random forest algorithm is high, it is not easy to interpret what is happening inside the forest because of the two-step randomization.

[What is the usefulness of analyzing the relationship between features and model performance?]

Understanding the role of features is worthwhile even if producing an accurate and generalisable model is the only objective of the modelling. This is because the less transparency of the model may be distrusted regardless of their predictive performance. The methodology we propose here is a novel application of machine learning interpretability methods to visualize and explore the role of features in forecast-model selection.

This paper proceeds as follows. In Section 2 we describe the application of FFORMS framework to M4competition data. The main contribution of this from our previous work Talagala, Hyndman & Athanasopoulos (2018) is we extend the FFORMS framework to model weekly, daily and hourly series.Section 3 gives background on machine learning interpretability techniques that are used to identify role of features in forecast model selection. In Section 4 we discuss the results. **??** concludes.

2 FFORMS Application to M4 competition data

The FFORMS framework consists of two main components: i) *offline phase*, which includes the development of a classification model and ii) *online phase*, use the classification model developed in the offline phase to identify "best" forecast-model. We develop separate classifiers for yearly, monthly, quarterly, weekly, daily and hourly series.

2.1 FFORMS framework: offline phase

2.1.1 observed sample

We split the time series in the M4 competition into training set and test set. The time series in the training set are used as the set of observed time series. The time series in the test set are used to evaluate the classification models. Further, for yearly, quarterly and monthly time series in addition to the time series provided in the M4 competition we used the time series of M1 and M3 competitions. Table 1 summarizes the number of time series in the observed sample and the test set in each frequency category.

Table 1: *Composition of the time series in the observed sample and the test set*

Frequency	Observed Sample			Test set
	M1	M3	M4	M4
Yearly	181	645	22000	1000
Quarterly	203	756	23000	1000
Monthly	617	1428	47000	1000
Weekly	-	-	259	100
Daily	-	-	4001	226
Hourly	-	-	350	64

2.1.2 simulated time series

As described in Talagala, Hyndman & Athanasopoulos (2018), we augment the reference set by adding multiple time series simulated based on each series in the M4 competition. We use several standard automatic forecasting algorithms to simulate multiple time series from each series. Table 2 shows the different automatic forecasting algorithms used under each frequency category. The automated ETS and ARIMA are implemented using `ets` and `auto.arima` functions available in the forecast package in R (Hyndman et al. 2018). The `stlf` function in the forecast package (Hyndman et al. 2018) is used to simulate multiple time series based on multiple seasonal decomposition approach. As shown in Table 2 we fit models to each time series in the M4 competition database from the corresponding algorithm and then simulate multiple time series from the selected models. Before simulating time series from daily and hourly series we convert the time series into multiple seasonal time series (msts) objects. For daily time series with length less 366 the frequency is set to 7 and if the time series is long enough to take more than a year (length > 366), the series is converted to a multiple seasonal time series objects with frequencies 7 and 365.25. For hourly series, if the series length is shorter than 168, frequency is set to 24, if the length of the series is greater than 168 and less than or equals to 8766 only daily and weekly seasonality are allowed setting the frequencies to 24 and 168. In this experiment the length of the simulated time series is set to be equal to: length of the training period specified in the M4 competition + length of the forecast horizon specified in the competition. For example, the series with id "Y13190" contains a training period of length 835. The length of the simulated series generated based on this series is equals to 841 (835+6).

Table 2: *Automatic forecasting algorithms used to simulate time series*

Algorithm	Y	Q	M	W	D	H
automated ETS	✓	✓	✓			
automated ARIMA	✓	✓	✓			
forecast based on multiple seasonal decomposition				✓	✓	✓

As illustrated in Talagala, Hyndman & Athanasopoulos (2018), the observed time series and the simulated time series form the reference to build our classification algorithm. Once we create the reference set for random forest training we split each time series in the reference set into training period and test period.

2.1.3 Input: features

The FFORMS framework operates on the features of the time series. For each time series in the reference set features are calculated based on the training period of the time series.

Table 3: *Time series features*

	Feature	Description	Y	Q/M	W	D/H
1	T	length of time series	✓	✓	✓	✓
2	trend	strength of trend	✓	✓	✓	✓
3	seasonality 1	strength of seasonality corresponds to frequency 1	-	✓	✓	✓
4	seasonality 2	strength of seasonality corresponds to frequency 2	-	-	-	✓
5	linearity	linearity	✓	✓	✓	✓
6	curvature	curvature	✓	✓	✓	✓
7	spikiness	spikiness	✓	✓	✓	✓
8	e_acf1	first ACF value of remainder series	✓	✓	✓	✓
9	stability	stability	✓	✓	✓	✓
10	lumpiness	lumpiness	✓	✓	✓	✓
11	entropy	spectral entropy	✓	✓	✓	✓
12	hurst	Hurst exponent	✓	✓	✓	✓
13	nonlinearity	nonlinearity	✓	✓	✓	✓
14	alpha	ETS(A,A,N) $\hat{\alpha}$	✓	✓	✓	-
15	beta	ETS(A,A,N) $\hat{\beta}$	✓	✓	✓	-
16	hwalpha	ETS(A,A,A) $\hat{\alpha}$	-	✓	-	-
17	hwbeta	ETS(A,A,A) $\hat{\beta}$	-	✓	-	-
18	hwgamma	ETS(A,A,A) $\hat{\gamma}$	-	✓	-	-
19	ur_pp	test statistic based on Phillips-Perron test	✓	-	-	-
20	ur_kpss	test statistic based on KPSS test	✓	-	-	-
21	y_acf1	first ACF value of the original series	✓	✓	✓	✓
22	diff1y_acf1	first ACF value of the differenced series	✓	✓	✓	✓
23	diff2y_acf1	first ACF value of the twice-differenced series	✓	✓	✓	✓
24	y_acf5	sum of squares of first 5 ACF values of original series	✓	✓	✓	✓
25	diff1y_acf5	sum of squares of first 5 ACF values of differenced series	✓	✓	✓	✓
26	diff2y_acf5	sum of squares of first 5 ACF values of twice-differenced series	✓	✓	✓	✓
27	seas_acf1	autocorrelation coefficient at first seasonal lag	-	✓	✓	✓
28	sediff_acf1	first ACF value of seasonally-differenced series	-	✓	✓	✓
29	sediff_seacf1	ACF value at the first seasonal lag of seasonally-differenced series	-	✓	✓	✓
30	sediff_acf5	sum of squares of first 5 autocorrelation coefficients of seasonally-differenced series	-	✓	✓	✓
31	lmres_acf1	first ACF value of residual series of linear trend model	✓	-	-	-
32	y_pacf5	sum of squares of first 5 PACF values of original series	✓	✓	✓	✓
33	diff1y_pacf5	sum of squares of first 5 PACF values of differenced series	✓	✓	✓	✓
34	diff2y_pacf5	sum of squares of first 5 PACF values of twice-differenced series	✓	✓	✓	✓

The description of the features calculated under each frequency category is shown in Table 3. A comprehensive description of the features used in the experiment is given in Talagala, Hyndman & Athanasopoulos (2018).

2.1.4 Output: class-labels

In addition to the class labels used by Talagala, Hyndman & Athanasopoulos (2018) we include some more class labels when applying the FFORMS framework to the M4 competition time series. The description of class labels considered under each frequency is shown in Table 4. We fit the corresponding models outlined in Table 4 to each series in the reference set. The models are estimated using the training period for each series, and forecasts are produced for the test periods.

The `auto.arima` and `ets` functions in the forecast package are used to identify the suitable

Table 4: *Class labels*

class label	Description	Y	Q/M	W	D/H
WN	white noise process	✓	✓	✓	✓
AR/MA/ARMA	AR, MA, ARMA processes	✓	✓	✓	-
ARIMA	ARIMA process	✓	✓	✓	-
SARIMA	seasonal ARIMA	✓	✓	✓	-
RWD	random walk with drift	✓	✓	✓	✓
RW	random walk	✓	✓	✓	✓
Theta	standard theta method	✓	✓	✓	✓
STL-AR		-	✓	✓	✓
ETS-notrendnoseasonal	ETS without trend and seasonal components	✓	✓	✓	-
ETStrendonly	ETS with trend component and without seasonal component	✓	✓	✓	-
ETSdampedtrend	ETS with damped trend component and without seasonal component	✓	✓	-	-
ETStrendseasonal	ETS with trend and seasonal components	-	✓	-	-
ETSdampedtrendseasonal	ETS with damped trend and seasonal components	-	✓	-	-
ETSseasonalonly	ETS with seasonal components and without trend component	-	✓	-	-
snaive	seasonal naive method	✓	✓	✓	✓
tbats	TBATS forecasting	-	✓	✓	✓
nn	neural network time series forecasts	✓	✓	✓	✓
mstlets		-	-	✓	✓
mstlarima		-	-	-	✓

(S)ARIMA and ETS models. In order to identify the "best" forecast-model for each time series in the reference set we combine the mean Absolute Scaled Error (MASE) and the symmetric Mean Absolute Percentage Error (MAPE) calculated over the test set. More specifically, for each series both forecast error measures MASE and sMAPE are calculated for each of the forecast models. Each of these is respectively standardized by the median MASE and median sMAPE calculated across the methods. The model with the lowest average value of the scaled MASE and scaled sMAPE is selected as the output class-label. Most of the labels given in Table 4 are self-explanatory labels. In STL-AR, mstlets, and mstlarima, first STL decomposition method applied to the time series and then seasonal naive method is used to forecast the seasonal component. Finally, AR, ETS and ARIMA models are used to forecast seasonally adjusted data respectively.

2.1.5 Train a random forest classifier

A random forest with class priors is used to develop the classifier. We build separate random forest classifiers for yearly, quarterly, monthly, weekly, daily and hourly time series. The wrapper function called `build_rf` in the `seer` package enables the training of a random forest and returns class labels("best" forecast-model) for each time series.

2.2 FFORMS framework: online phase

The online phase of the algorithm involves generating point forecasts and 95% prediction intervals for the M4 competition data. First, the corresponding features are calculated based on the full length of the training period provided by the M4 competition. Second, point forecasts and 95% prediction intervals are calculated based on the predicted class labels, in this case

forecast-models. Finally, all negative values are set to zero.

3 Machine Learning Interpretability

In recent years, there have been a growing interest for interpretability of machine learning algorithms with European General Data Projection Regulation (GDPR) stipulates the explainability of all automatically made decision concerning individuals. We explore the role of features in three different angles: i) global interpretability, and ii) local interpretability. We will introduce each of these ideas briefly below. Model-diagnostics tools are used.

3.1 General Notation

Let $\mathcal{P} = \{(\mathbf{x}^{(\mathbf{i})}, y^{(i)})\}_{i=1}^{N}$ be the historical data set we use to train the classifier. Consider a p-dimensional feature vector $X = (X_1, X_2, ..., X_p)$ and a dependent variable, best forecasting method for each series Y. Let $\mathcal{F}$ be the unkown relationship between X and Y. Zhao & Hastie (n.d.) term this as "law of nature". Inside the FFORMS framework, random forest algorithm tries to learn this relationship using the historical data we provided. We denote the predicted function as $\hat{\mathcal{F}}$.

3.2 Global Interpretability Methods

Global interpretability evaluate the behavior of a model on entire data set. Global perspective of model interpretation helps users to understand the overall modeled relationship between features and the model outcome. For example, which features are contribute mostly to the predictive mechanism of the fitted model, complex interactions between features, etc. In the following subsections we provide a description of tools we use to explore the global perspective of the model.

3.2.1 Permutation-based variable importance measure

The permutation-based variable importance introduced by Breiman (2001) measures the the prediction strength of each feature. This measure is calculated based on the out-of-bag (OOB) observations. The calculation of variable importance is formalized as follow: Let $\bar{\mathcal{B}}^{(k)}$ be the OOB sample for a tree k, with $k \in \{1, ..., ntree\}$, where $ntree$ is the number of trees in the random forest. Then the variable importance of variable X_j in k^{th} tree is:

$$VI^{(k)}(X_j) = \frac{\sum_{i \in \bar{\mathcal{B}}^{(k)}} I(\gamma_i = \gamma_{i,\pi_j}^{k})}{|\bar{\mathcal{B}}^{(k)}|} - \frac{\sum_{i \in \bar{\mathcal{B}}^{(k)}} I(\gamma_i = \gamma_i^{k})}{|\bar{\mathcal{B}}^{(k)}|},$$

where γ_i^k denotes the predicted class for the i^{th} observation before permuting the values of X_j and γ_{i,π_j}^k is the predicted class for the i^{th} observation after permuting the values of X_j. The

overall variable importance score is calculated as:

$$VI(X_j) = \frac{\sum_1^{ntree} VI^{(t)}(x_j)}{ntree}.$$

Permutation-based variable importance measures provide a useful starting point for identifying relative influence of features on the predicted outcome. However, they provide a little indication of the nature of the relationship between the features and model outcome. To gain further insights into the role of features inside the FFORMS framework we use partial dependence plot (PDP) introduced by Friedman, Popescu, et al. (2008).

3.2.2 Partial dependence plot (PDP)

Partial dependence plot can be used to graphically examine how each feature is related to the model prediction while accounting for the average effect of other features in the model. This method is particularly useful to discover complex model structures in "black box" models. Let X_s be the feature we want examine partial dependencies for and X_c be the remaining set of features in X. Then f_s, the partial dependence function on x_s is defines as {f_s=E_x}

3.2.3 Variable importance measure based on PDP

3.2.4 ICE plots

While partial dependence plots. However, in the presence of substantial interaction between features, the partial dependence response curves can be misleading. ICE curves are an extension of PD-plots but, rather than plot the average marginal effect on the response variable, we plot the change in the predicted response variable for each observation as we vary each feature.

3.2.5 Variable importance measure based on ICE curves

PD plots, ICE curves and PD-, ICE-associated measures are computationally intensive to create, especially when there are large number of observations in the training set. Hence those measures are calculated based on the subset of randomly selected training examples.

Many of the other techniques exist for visualizing the relationship between the predictors and the response based on a fitted model. For example,

3.3 Representation of model in the data space (m-in-ds) and data in the model space (d-in-ms)

3.4 Local Interpretability Methods

Global interpretations help us to understand entire modeled relationship. Local interpretations help us understand the predictions of the model for a single instance or a group of similar instances.

3.5 Model-diagnostics

In addition to the previews discussions xx propose a novel application of

3.5.1 Out-of-bag(OOB) error and uncertainty measure for each observation

It is argued in order to estimate the test error of a bagged model it is not necessary to perform cross-validation approach, because each tree is grown using different bootstrap samples from the training set and a part of training data is not used in the tree construction [ref]. In general, each bagged tree does not make use of around one third of observations to construct the decision tree. These observations are referred to as the out-of-bag(OOB) observations. Each tree is grown based on different bootstrap samples hence, each tree has different set of OOB observations. These OOB samples can be used to calculate internal estimation of the test set error, which is known as OOB error.

4 Results

4.1 M4 competition

4.2 Machine learning interpretability

In this section we provide a step-by-step description on how machine leaning interpretability tools are used to facilitate better understanding of of FFORMS framework and corresponding results. We provide separate descriptions for yearly, quarterly, monthly, weekly, daily and hourly series.

4.2.1 Yearly data

Model diagnostic: FFORMS framework,Yearly series

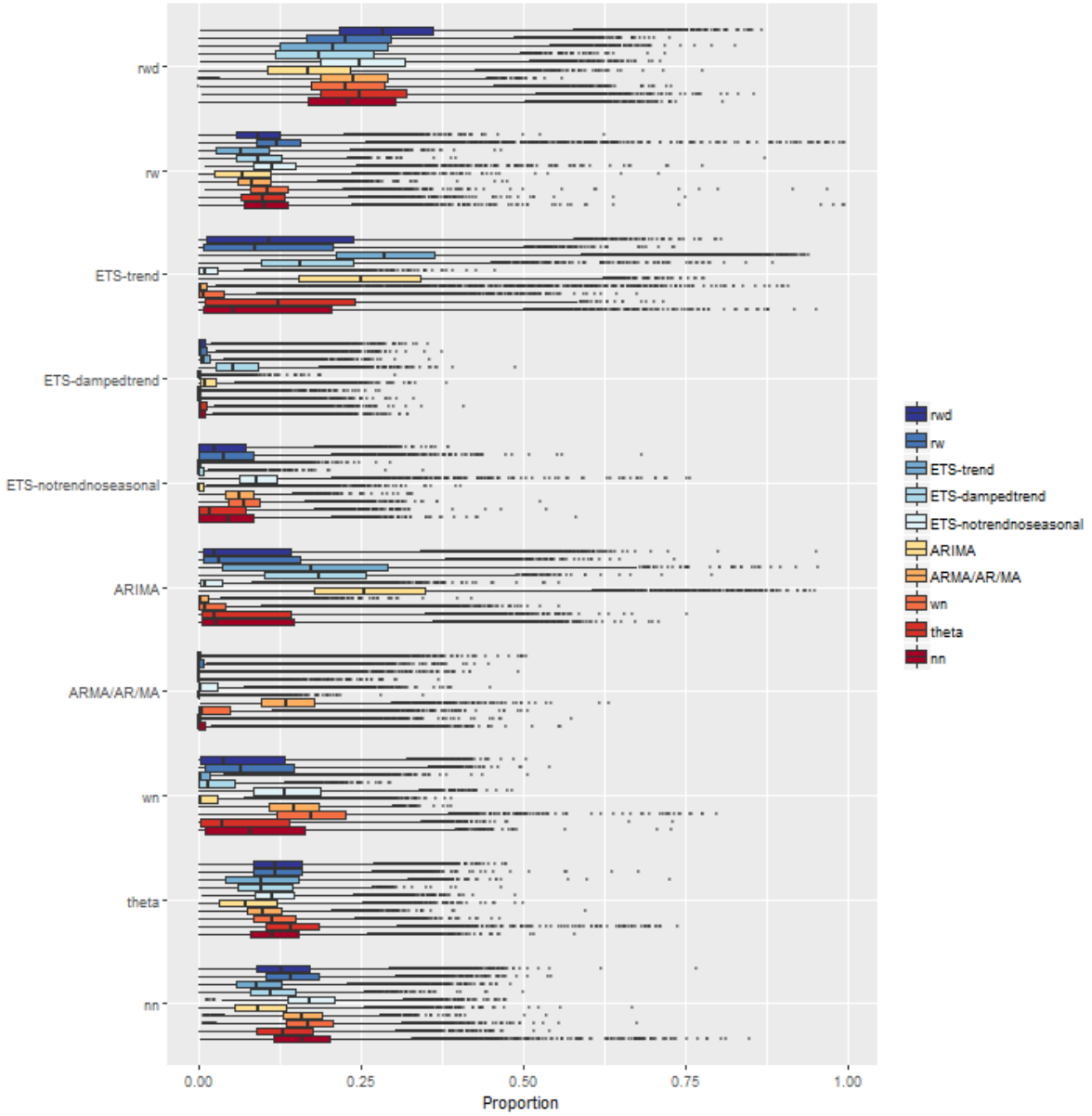


Figure 1: *(#fig:yearly_oob)Distribution of proportion of times each time series was assigned to each class in the forest. Each row represent the predicted class label and colour of boxplots corresponds to true class label. There are ten rows in the plot corresponds to each predicted class represented by Y-axis. X-axis denotes the proportion of times a time series is classified in each class. On each row, the true class label match with the predicted class label category dominated the top, indicating a fairly good classification of the model fitted.*

?? shows the distribution of proportion of times each observation (in our case each time series) was assigned to each class based on OOB sample. Each row represent the **predicted class label** and colour of boxplots corresponds to **true class label**. The proportion 1 indicates, that the time series was always predicted to the corresponding class and 0 being never. This is an alternative way of visualizing the vote-matrix information in the random forest model. The other way of representing vote matrix involves ternary plot (Sutherland et al. (2000)) and jittered

side-by-side dotplot (Ehrlinger 2015; Silva, Cook & Lee 2017). To over come the problem of overlapping classes arises due to the scale of the training data set, number of classes and similar types of classes of data, boxplot diagrams are used. On each row of **??** the distribution of proportions corresponds to the time series in which the prefdictied class label and true class label are the same dominates the top indicating a fairly good classification of the model. In addition to that, in each row the distributions corresponds to the time series labeled similar to the predicted class label also dominate the others. For example, within ETS-trend predicted class, the distributions correspond to the true class labels, ETS-damped trend, ARIMA, were also assigned with high probability and less values were assigned to ARMA/AR/MA, White noise process and ETS(ANN)/ETS(MNN). This confirms that our FFORMS framework successfully learnt the similarities and dissimilarities between the classes itself. (Each series is classified as random walk in some instances, yearly time series has high chance of getting classified as random walk). One reason for this is as shown in Kang, Hyndman, Li, et al. (2018) yearly series of M1, M3 and M4 are generally trended. (Results of M3 and M4 competition) This diagram helps to evaluate the model performance in the data space (model-in-the-data-space) (Silva, Cook & Lee (2017)).

Feature importance and main effects

Permutation-based variable importance and Gini feature importance measure are used to evaluate the overall feature importance. Moreover, most important features for each class is identified based on three measures: i) permutation-based variable importance, ii) partial dependence functions based variable importance and iii) ICE-curves based variable importance measure. The one that shows the highest importance is ranked 25, the second best is ranked 24, and so on. Finally, for each category, an average rank for each feature is computed based on the mean value of all rankings across all the feature importance metrics considered. The corresponding results are shown in **??**. The features strength of trend and test statistic of Phillips–Perron(PP) unit root test, linearity, first autocorrelation coefficient of the differenced series are appear to be most important features in each class. On the otherhand, sum of squares of first five autocorrelation coefficints of the twice-difference series and lumpiness show lowest contribution across many classes. The length of time series (N) is assigned a high importance in neural-network class compared to others. This could be due to neural network approach may be beneficial in forecasting time series with long history of observations. Furthur, first autocoreelation coefficient of the twice-differenced series is appear to be most important in ARIMA class as this category contains the higher order differenced series. Hurst exponent and entropy appear to be equally important in stationary classes. Within ETS-damped trend category

beta and curvature ranked as important features.

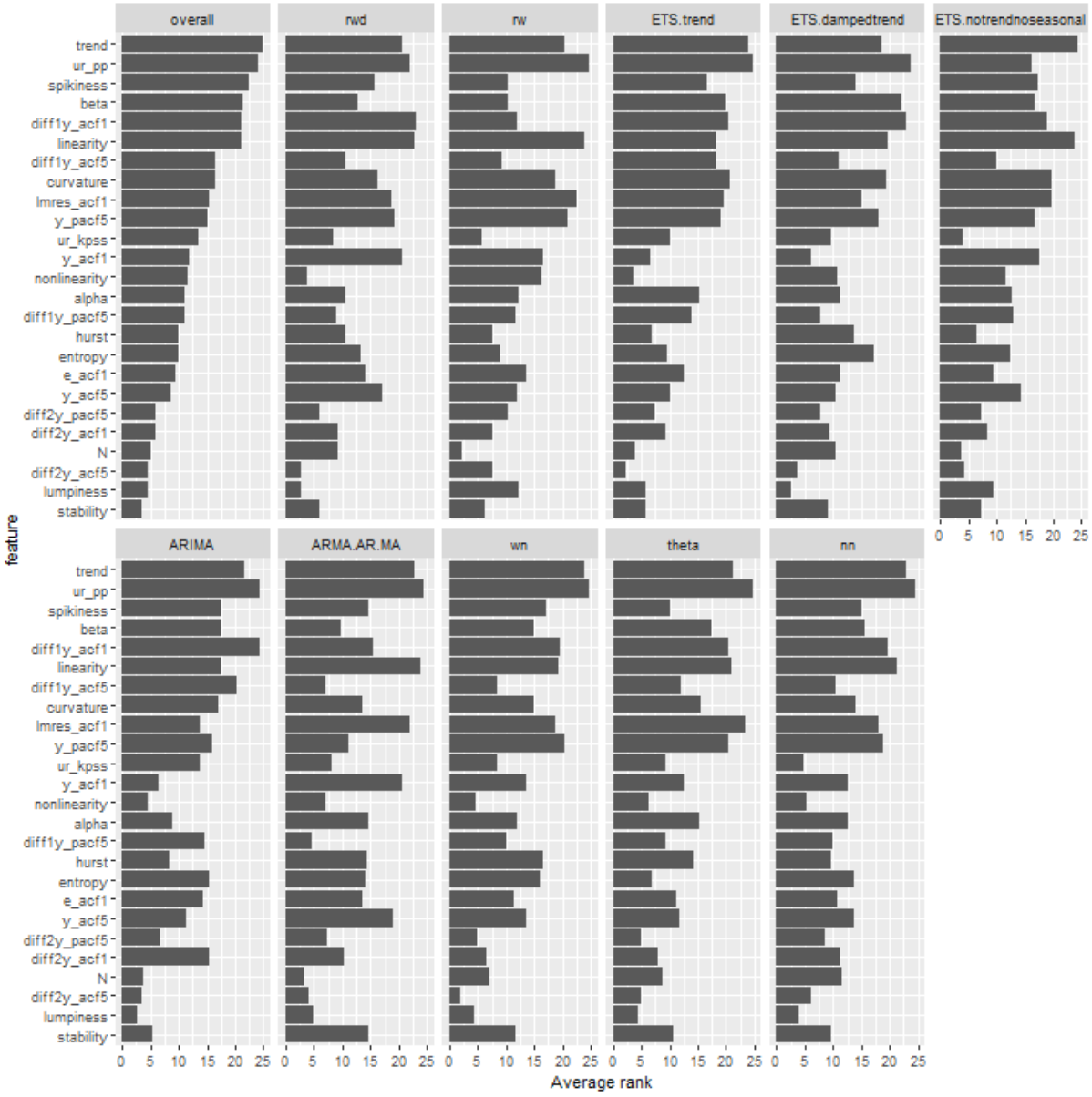


Figure 2: *(#fig:vi_yearly)Feature importance plot. Longer bars indicate more important features.*

Partial dependency curves

Partial dependency, and associated confidence intervals corresponds to top-three features of each category are plotted to explore the relationship between features and predicted outcome. Confidence intervals also facilitate the indication of interaction effects. Except, theta class monotonicall increasing or decresing relationship can be observed with trend. ,whereas the random walk with drift shows an opposite relationship.

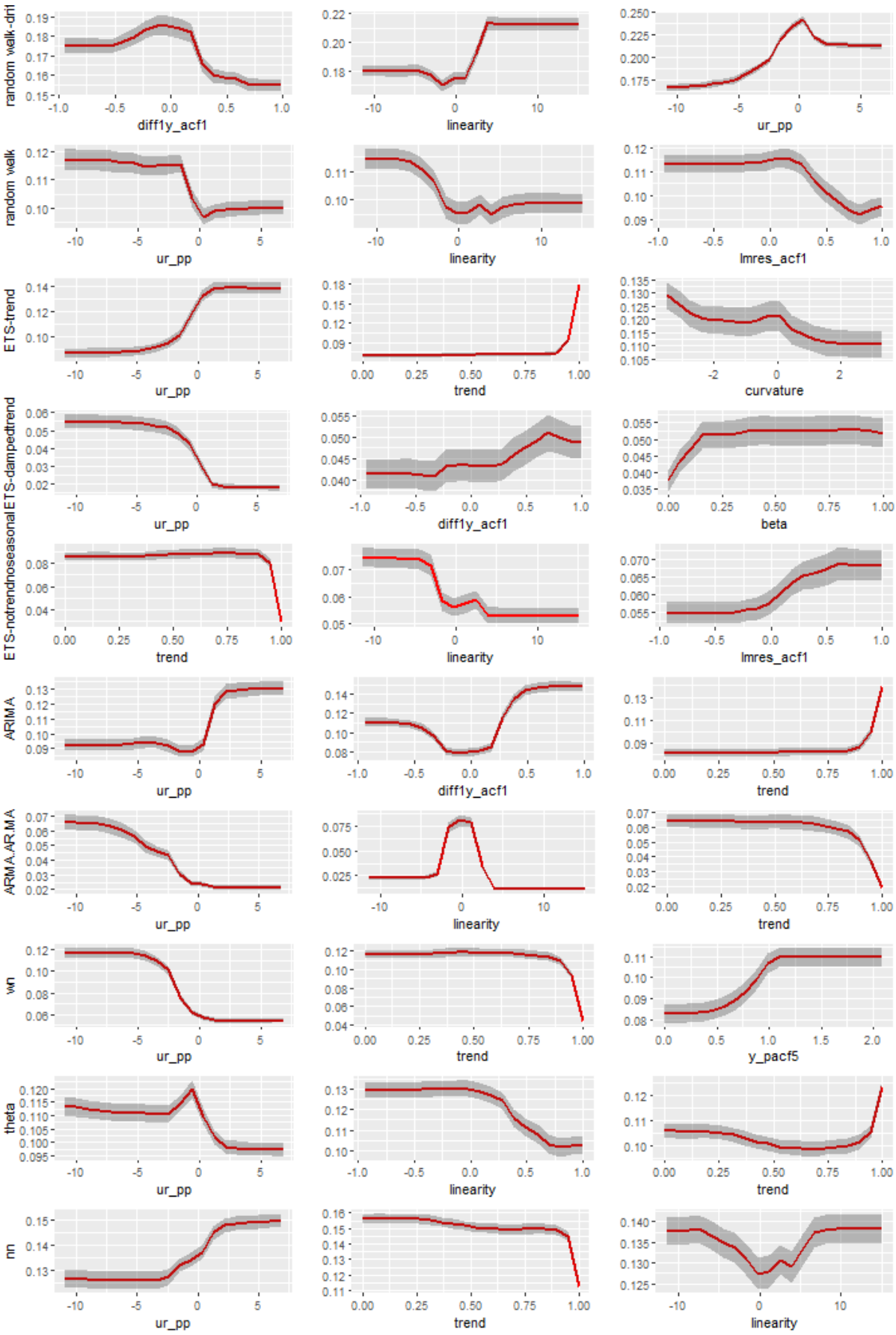
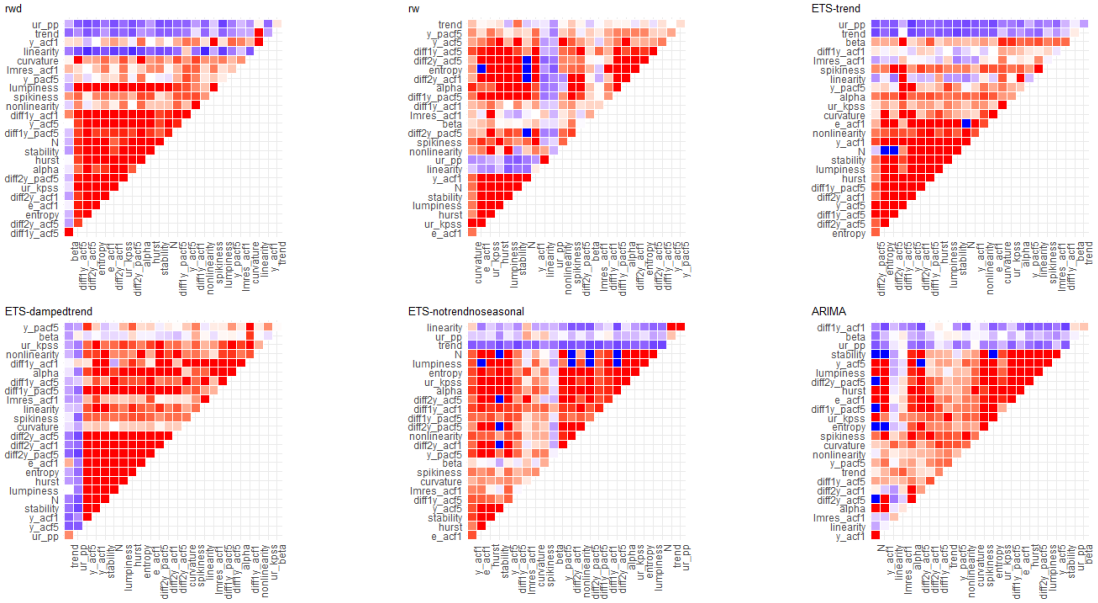


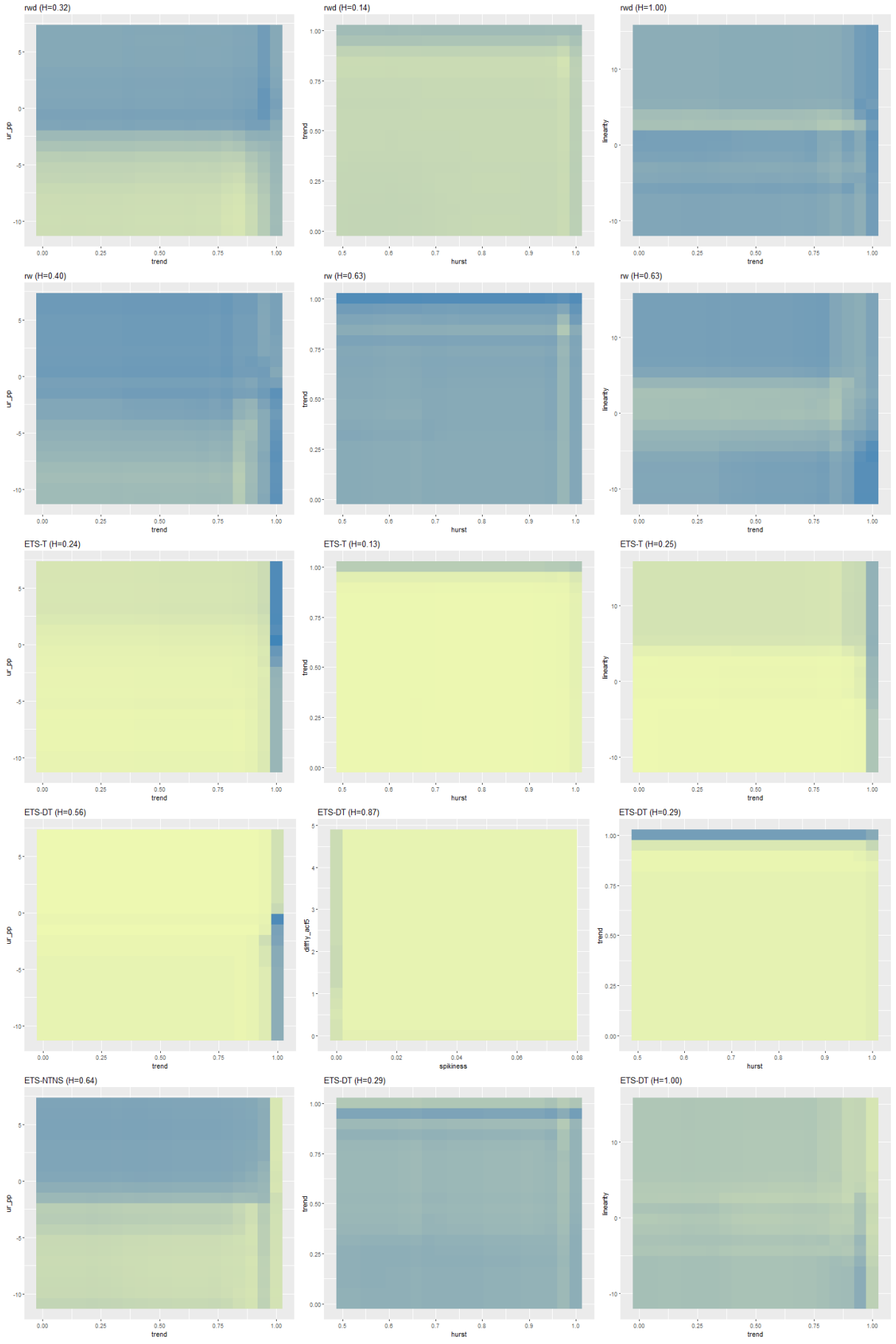
Figure 3: *(#fig:pdp_yearly)Partial dependence plots showing the variation top three features corresponds to each class. The shading shows the 95% confidence intervals. Y-axis denotes the probability of belong to corresponding class.*

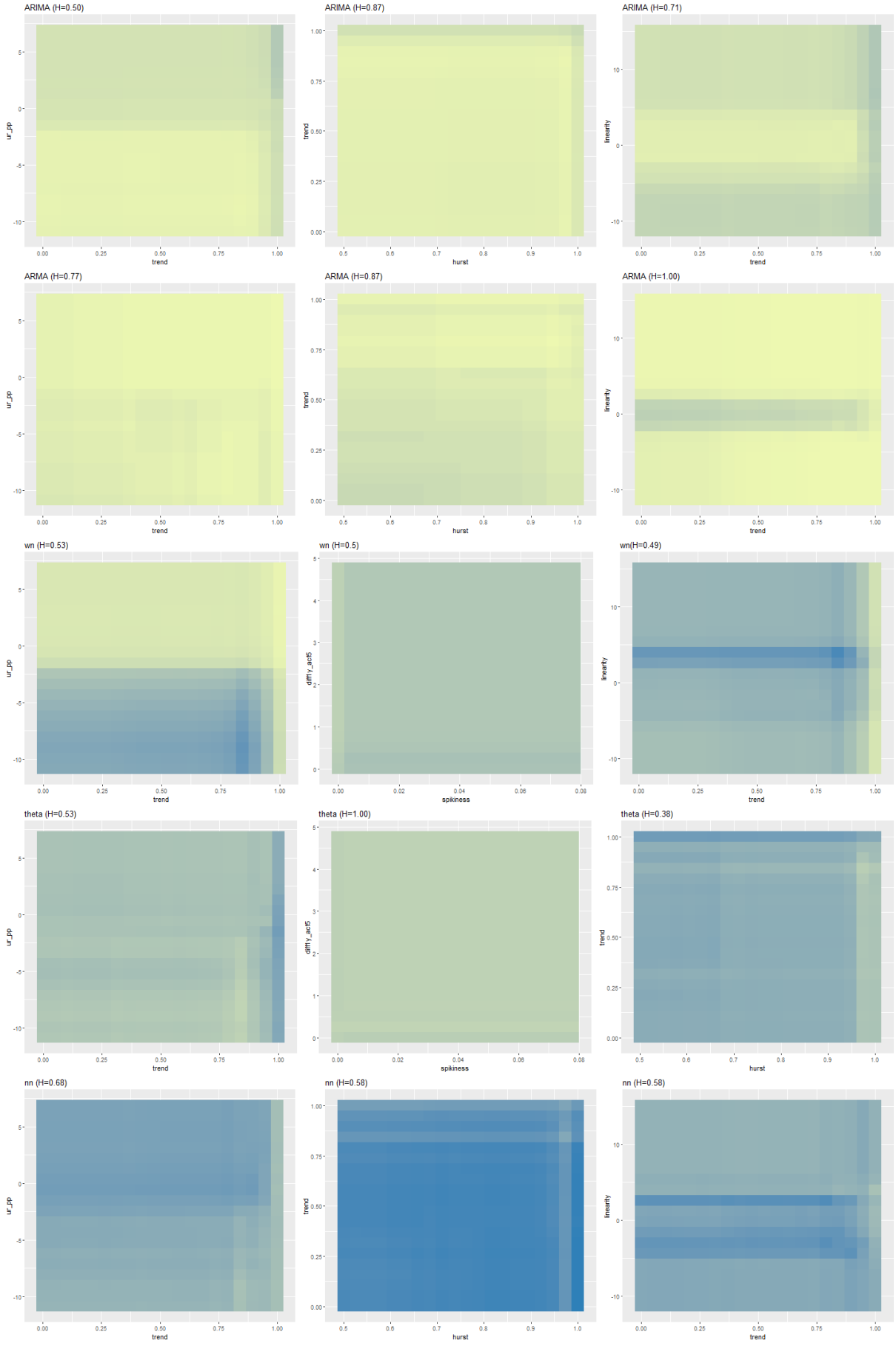
Two-way interaction between features

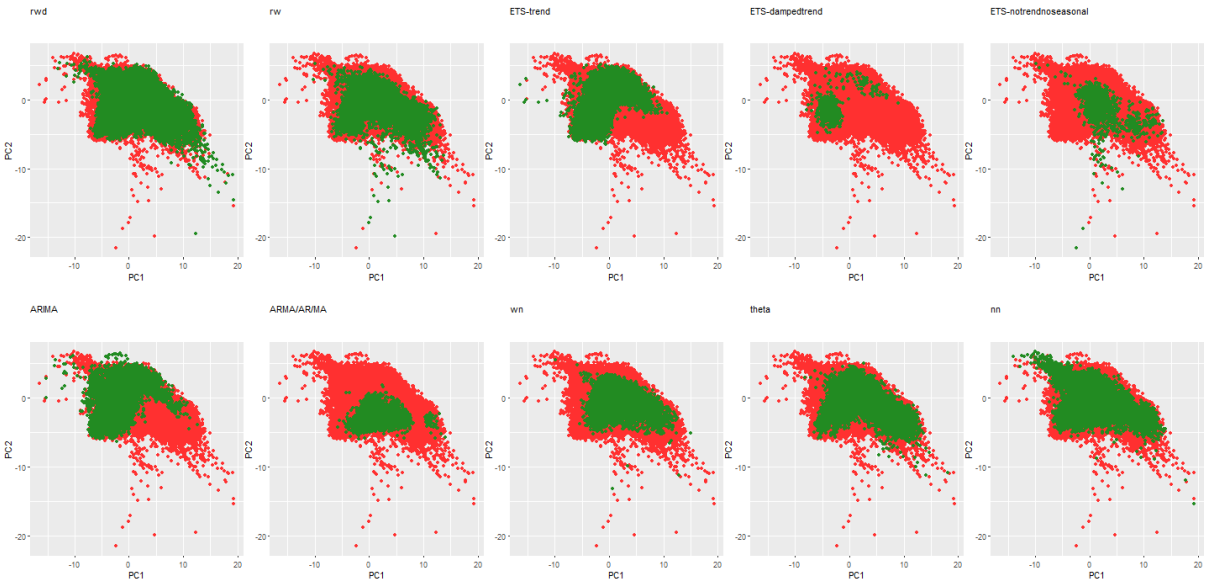
The relative strength of two-way interaction between features were determined using formulae developed by Friedman, Popescu, et al. (2008), which is implemented in the iml(Molnar, Casalicchio & Bischl (2018),) package in R. xx shows the heat maps of relative strength of all possible pairwise interactions for each class.



4.2.1.1 Two-way interaction effect between features



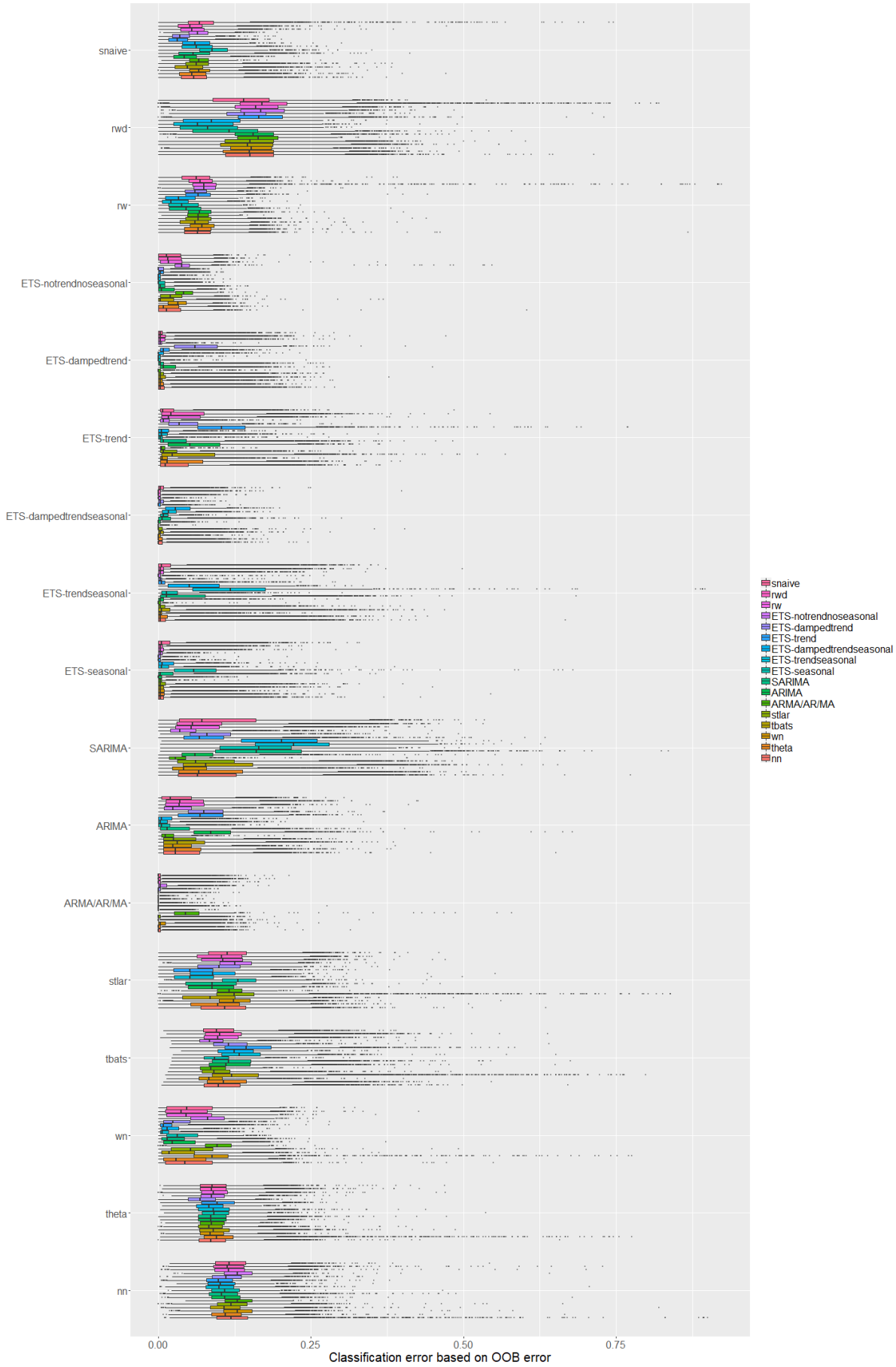

ARIMA (H=0.50)
ARIMA (H=0.87)
ARIMA (H=0.71)
ARMA (H=0.77)
ARMA (H=0.87)
ARMA (H=1.00)
wn (H=0.53)
wn (H=0.5)
wn(H=0.49)
theta (H=0.53)
theta (H=1.00)
theta (H=0.38)
nn (H=0.68)
nn (H=0.58)
nn (H=0.58)
ur_pp
trend
hurst
linearity
diff1y_acf5
spikiness

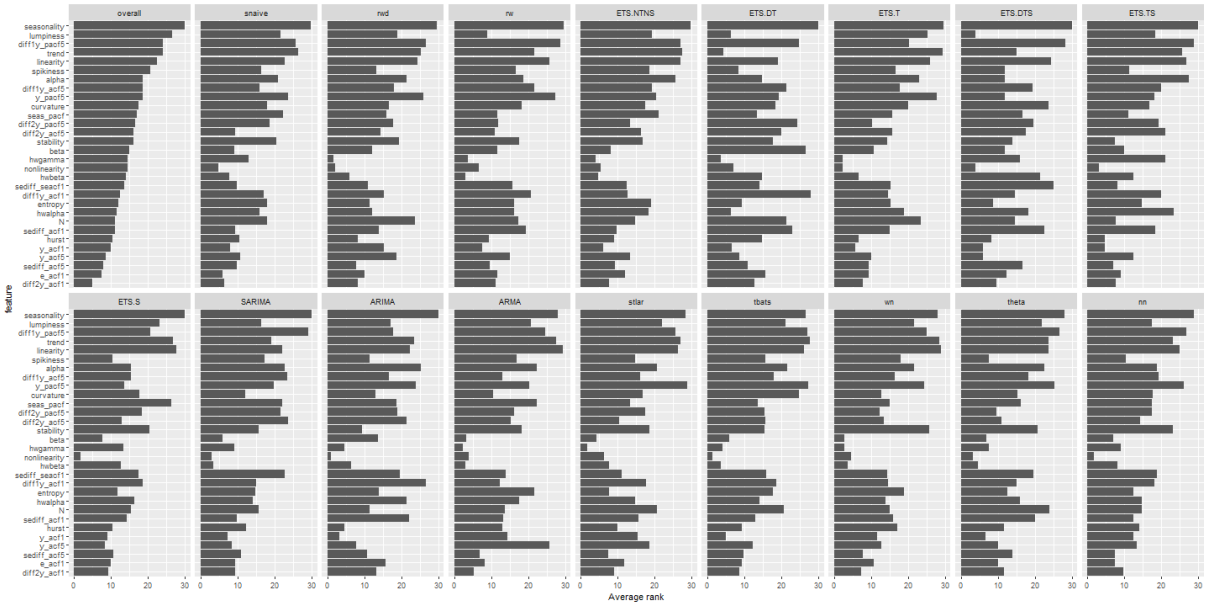


4.2.1.2 PCA space

4.2.1.3 LIME

4.2.2 Quarterly data




overall
snaive
rwd
rw
ETS.NTNS
ETS.DT
ETS.T
ETS.DTS
ETS.TS
ETS.S
SARIMA
ARIMA
ARMA
stlar
tbats
wn
theta
nn
feature
Average rank
0
10
20
30

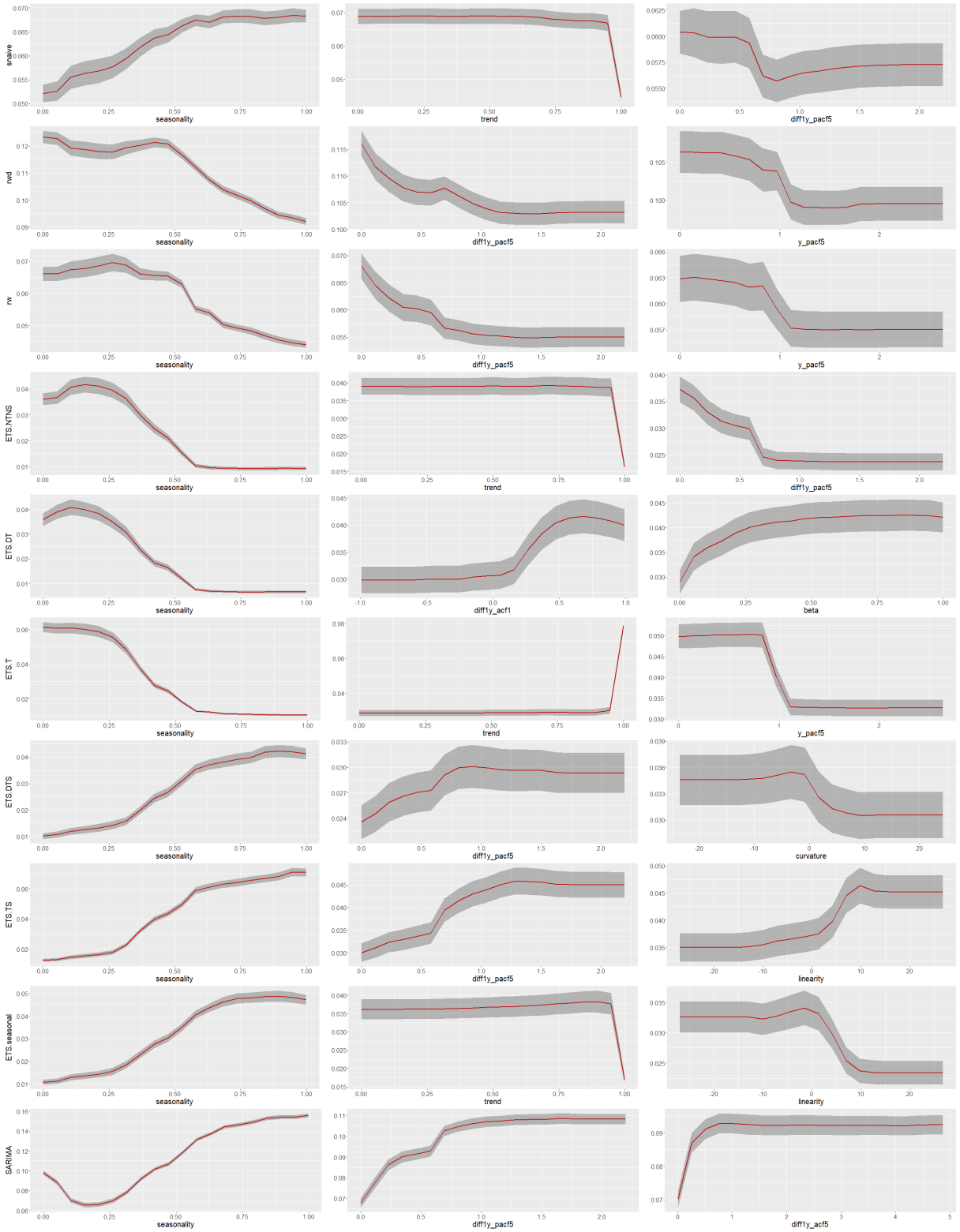


Figure 4: *(#fig:pdp_quarterly1)Partial dependence plots for top three features in each category (Quarterly)*

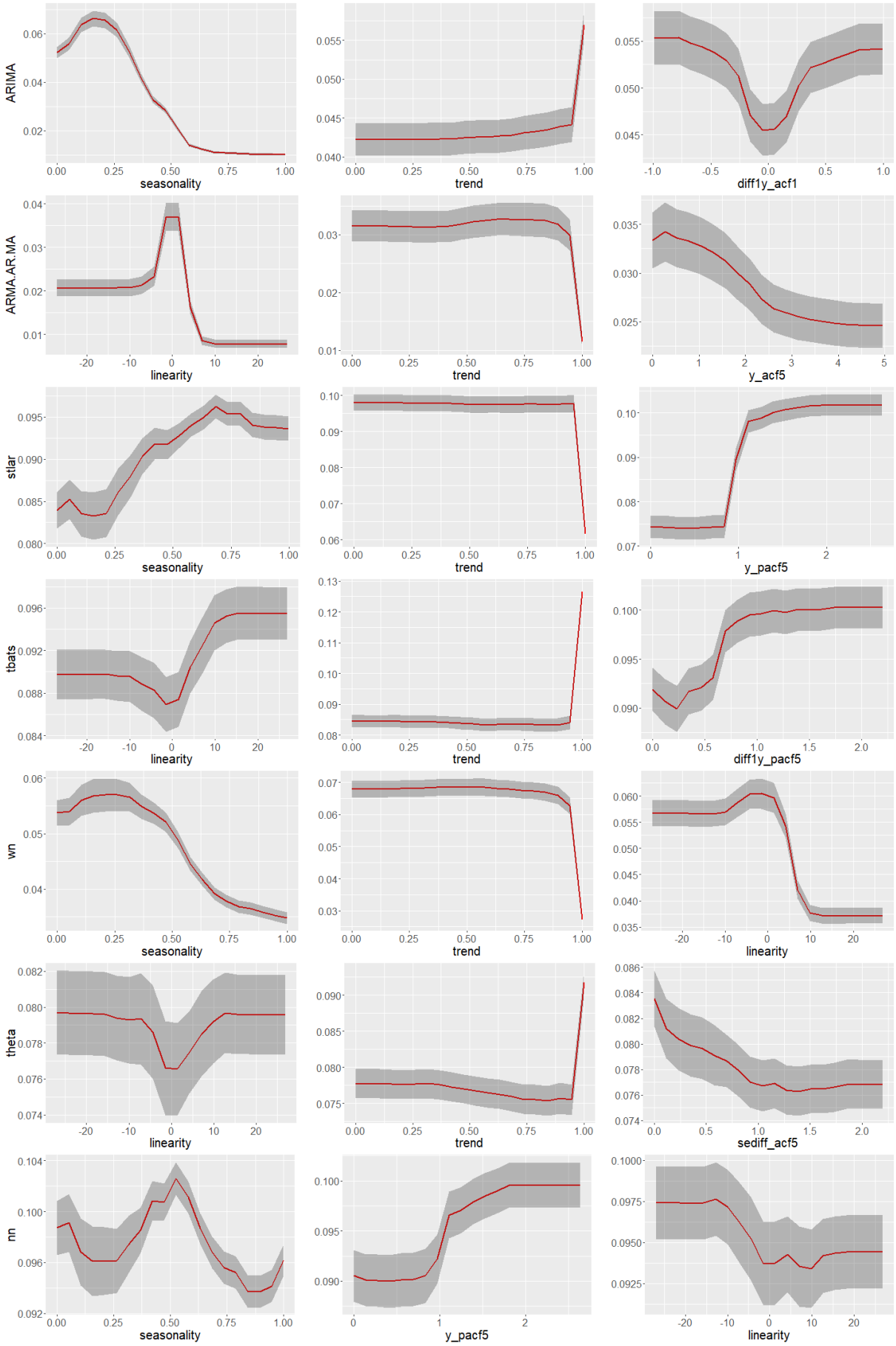
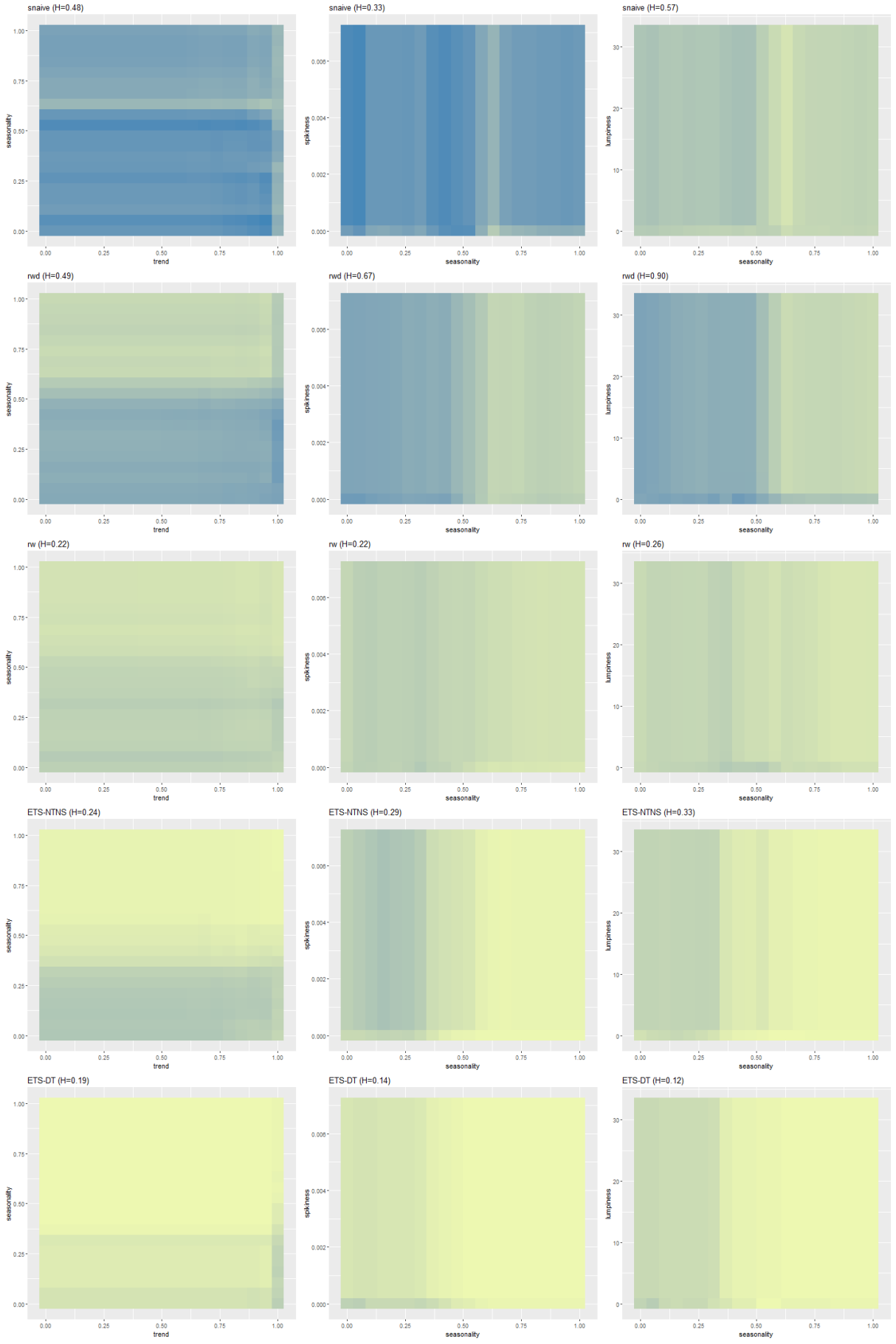
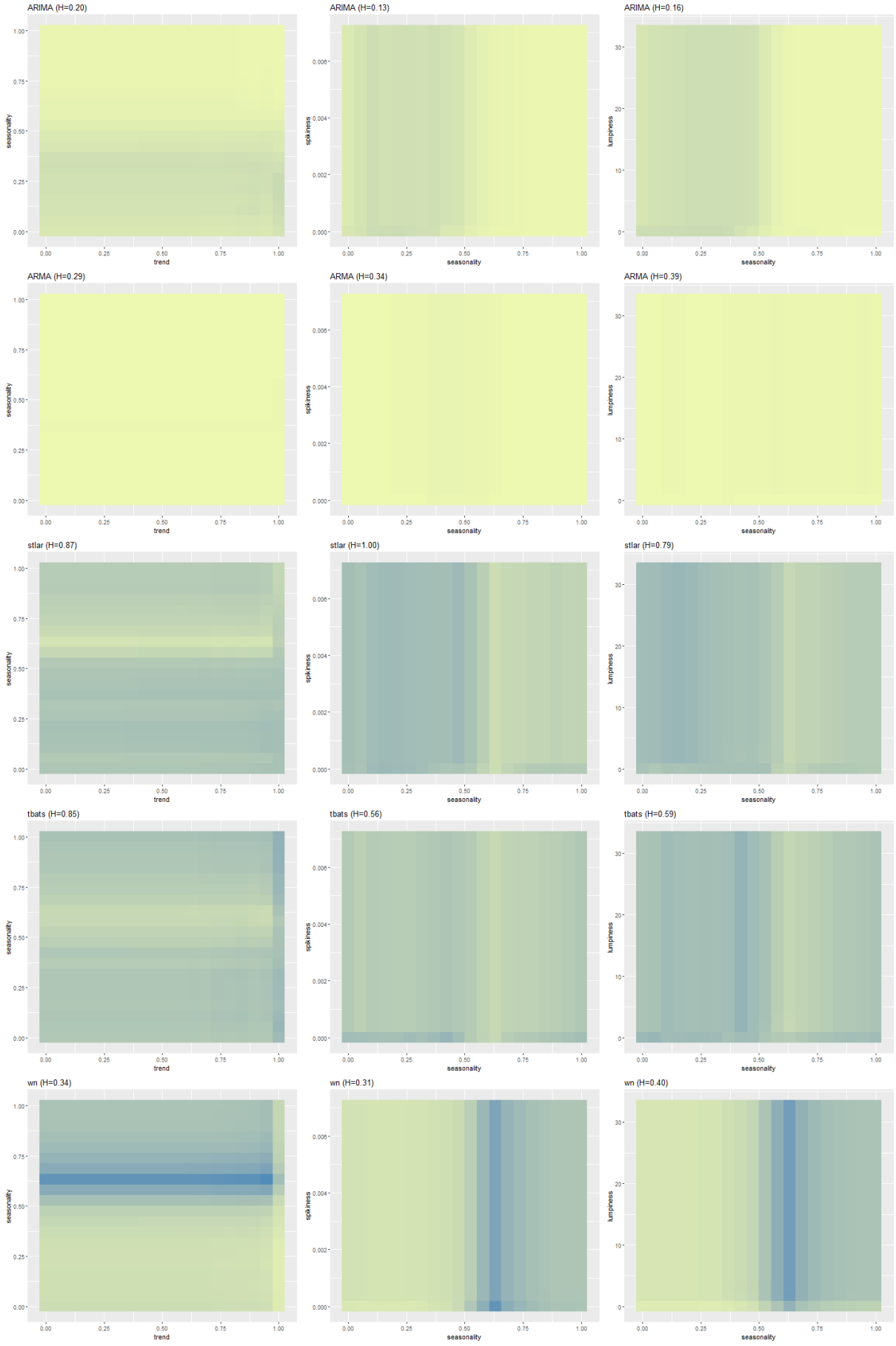


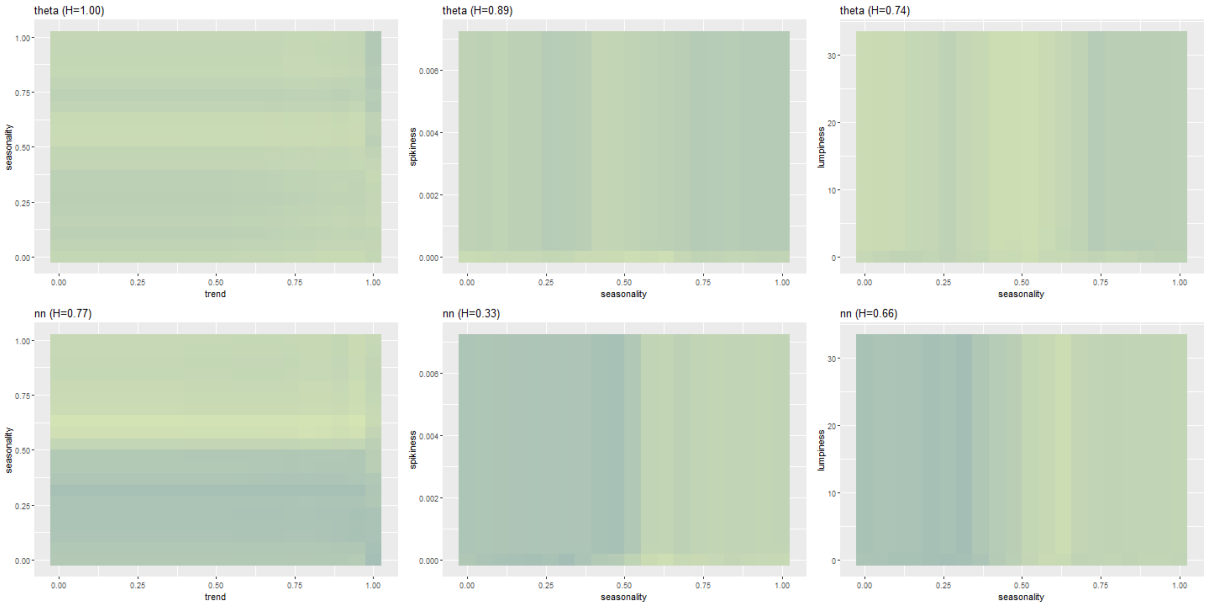
Figure 5: *(#fig:pdp_quarterly2)Partial dependence plots for top three features in each category (Quarterly)*

4.2.2.1 Two-way interaction effect between features




ETS-Trend (H=0.34)
ETS-Trend (H=0.09)
ETS-Trend (H=0.13)
ETS-DTS (H=0.49)
ETS-DTS (H=0.36)
ETS-DTS (H=0.25)
ETS-TS (H=0.49)
ETS-TS (H=0.37)
ETS-TS (H=0.31)
ETS-S (H=0.36)
ETS-S (H=0.40)
ETS-S (H=0.56)
SARIMA (H=0.18)
SARIMA (H=0.21)
SARIMA (H=0.15)
seasonality
trend
spikiness
seasonality
linearity
seasonality


ARIMA (H=0.20)
ARIMA (H=0.13)
ARIMA (H=0.16)
ARMA (H=0.29)
ARMA (H=0.34)
ARMA (H=0.39)
stlar (H=0.87)
stlar (H=1.00)
stlar (H=0.79)
tbats (H=0.85)
tbats (H=0.56)
tbats (H=0.59)
wn (H=0.34)
wn (H=0.31)
wn (H=0.40)
seasonality
spikiness
lumpiness
trend
seasonality


theta (H=1.00)
seasonality
trend
theta (H=0.89)
spikiness
seasonality
theta (H=0.74)
lumpiness
seasonality
nn (H=0.77)
seasonality
trend
nn (H=0.33)
spikiness
seasonality
nn (H=0.66)
lumpiness
seasonality

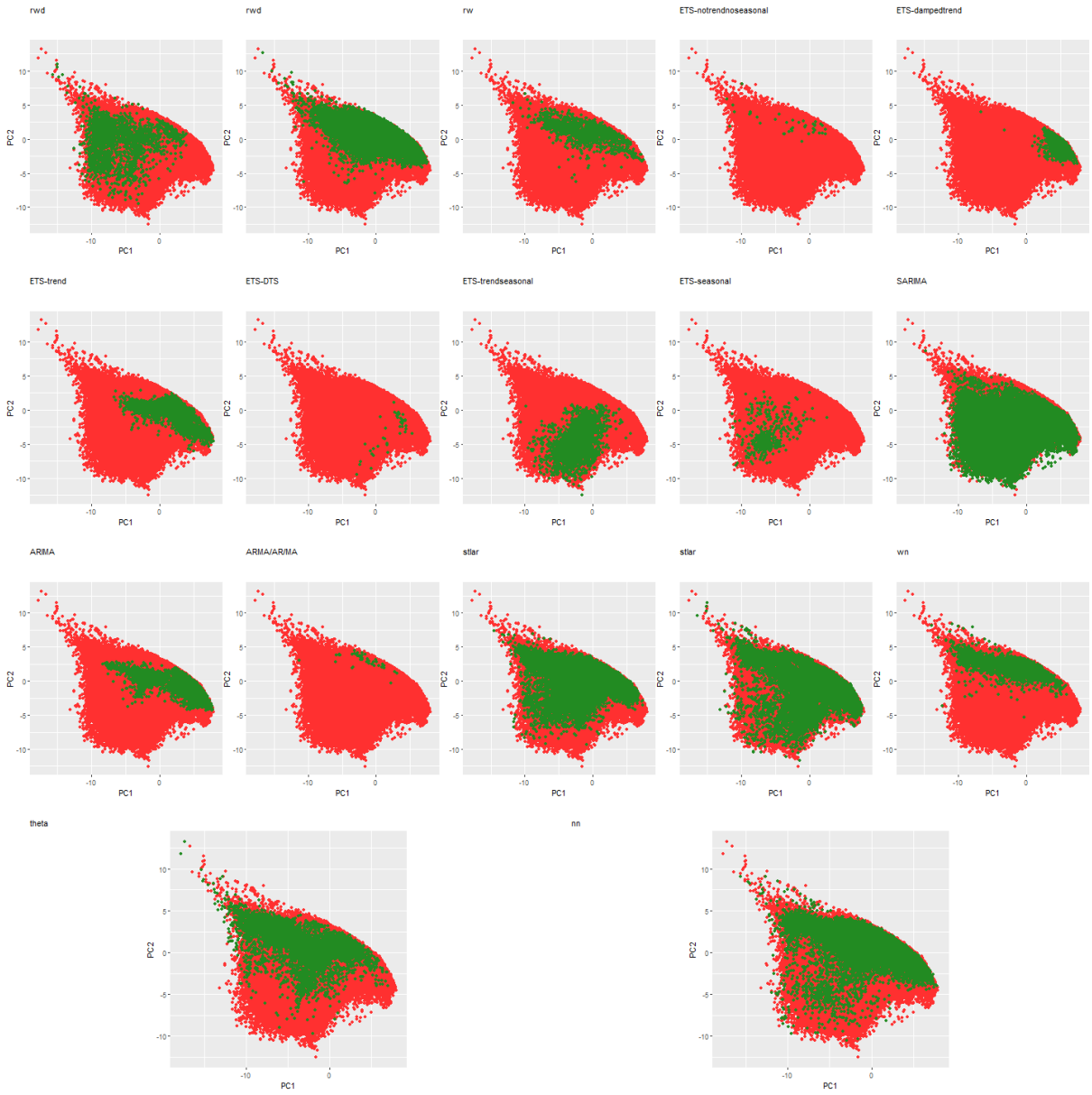
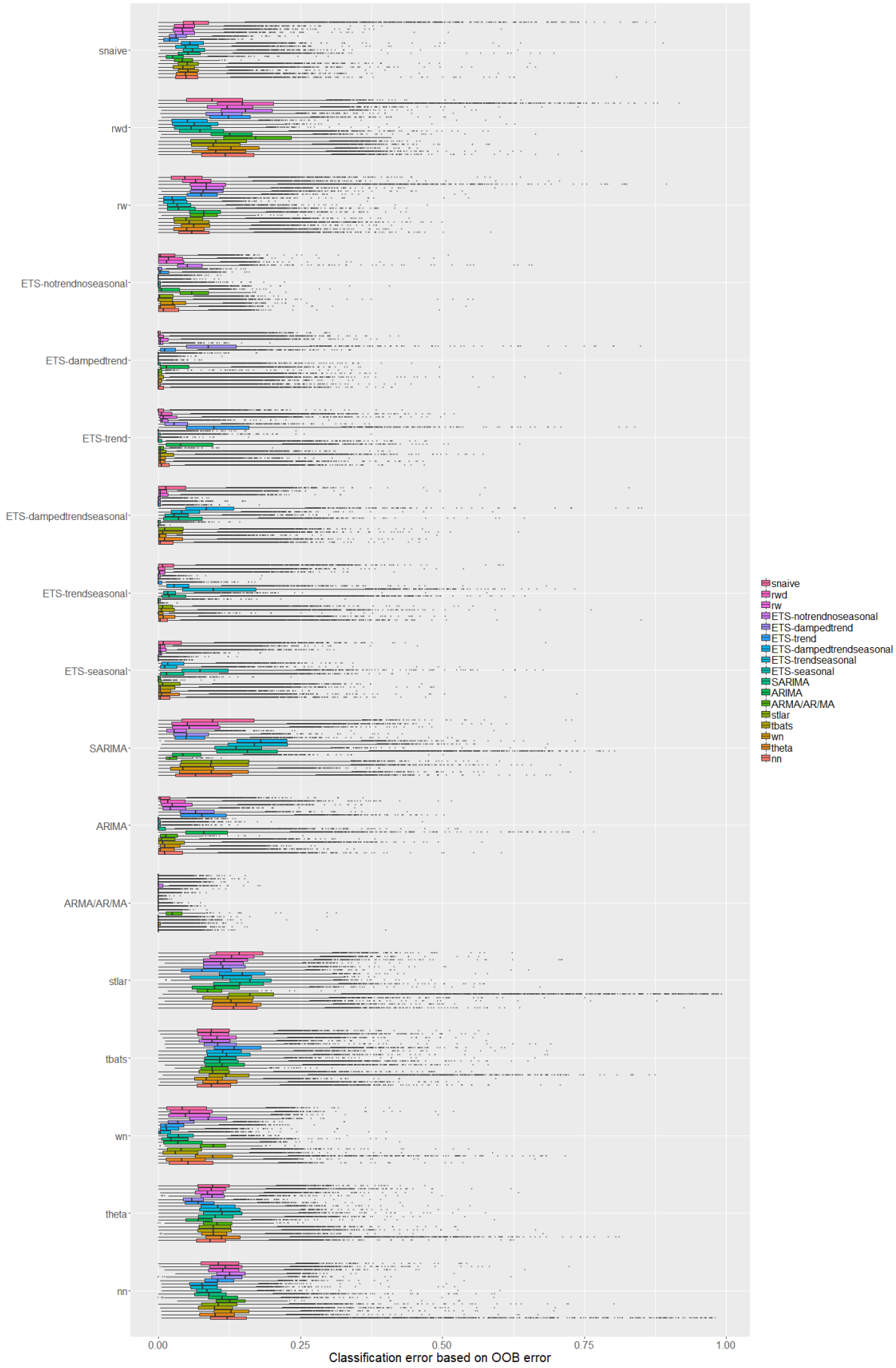
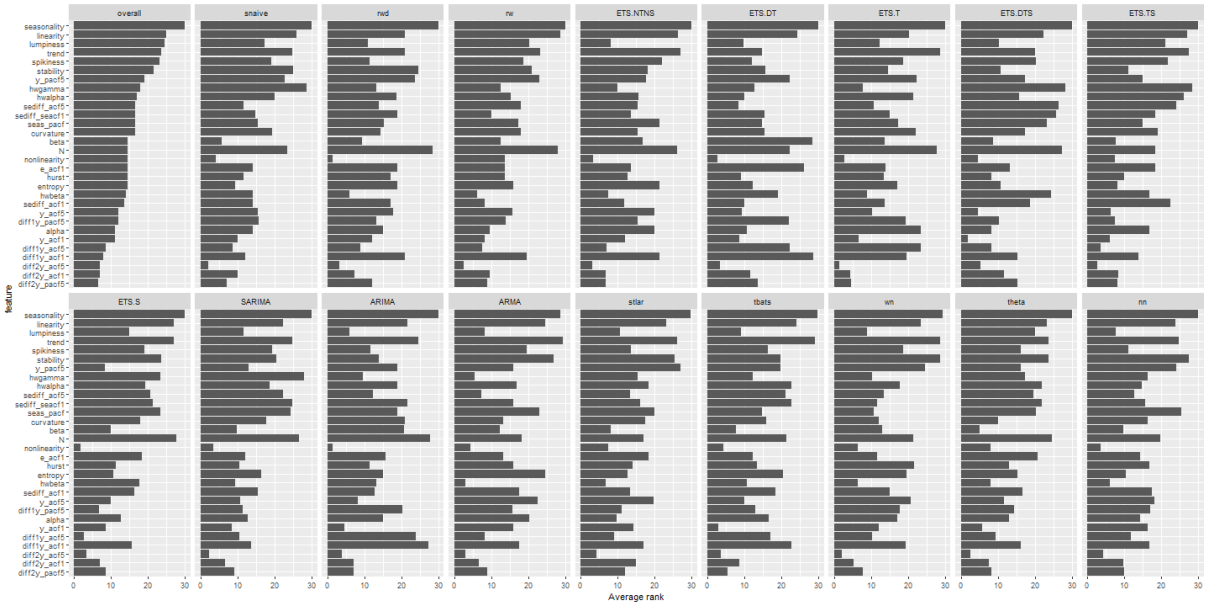


Figure 6: *(#fig:quarterly_pca)PCA(Quarterly)*

4.3 Monthly


snaive
rwd
rw
ETS-notrendnoseasonal
ETS-dampedtrend
ETS-trend
ETS-dampedtrendseasonal
ETS-trendseasonal
ETS-seasonal
SARIMA
ARIMA
ARMA/AR/MA
stlar
tbats
wn
theta
nn
0.00
0.25
0.50
0.75
1.00
Classification error based on OOB error


overall
snaive
rwd
rw
ETS.NTNS
ETS.DT
ETS.T
ETS.DTS
ETS.TS
ETS.S
SARIMA
ARIMA
ARMA
stlar
tbats
wn
theta
nn
feature
seasonality
linearity
lumpiness
trend
spikiness
stability
y_pacf5
hwgamma
hwalpha
sediff_acf5
sediff_seacf1
seas_pacf
curvature
beta
N
nonlinearity
e_acf1
hurst
entropy
hwbeta
sediff_acf1
y_acf5
diff1y_pacf5
alpha
y_acf1
diff1y_acf5
diff1y_acf1
diff2y_acf5
diff2y_acf1
diff2y_pacf5
0
10
20
30
Average rank

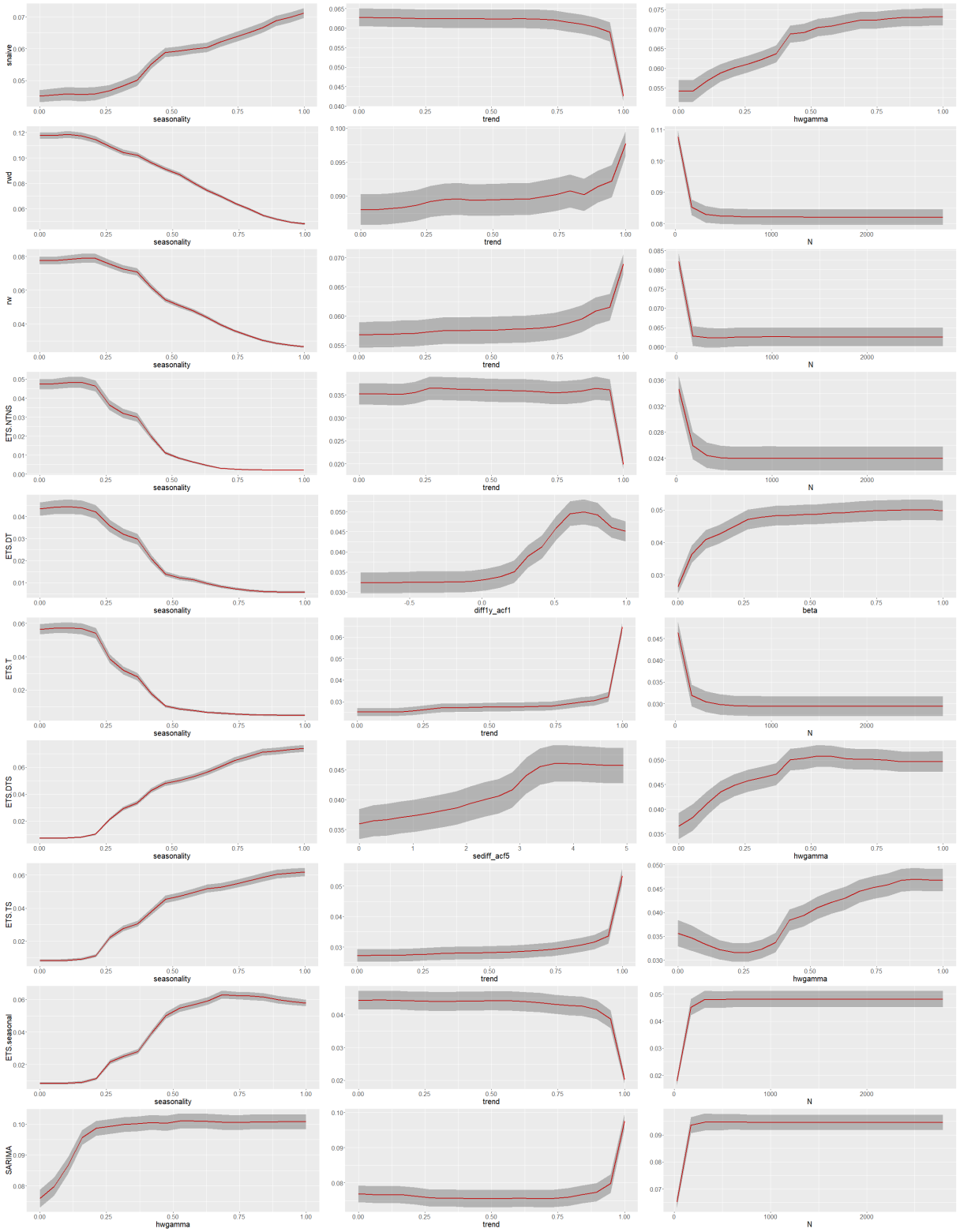


Figure 7: *(#fig:pdp_monthly1)Partial dependence plots for top three features in each category (Monthly)*

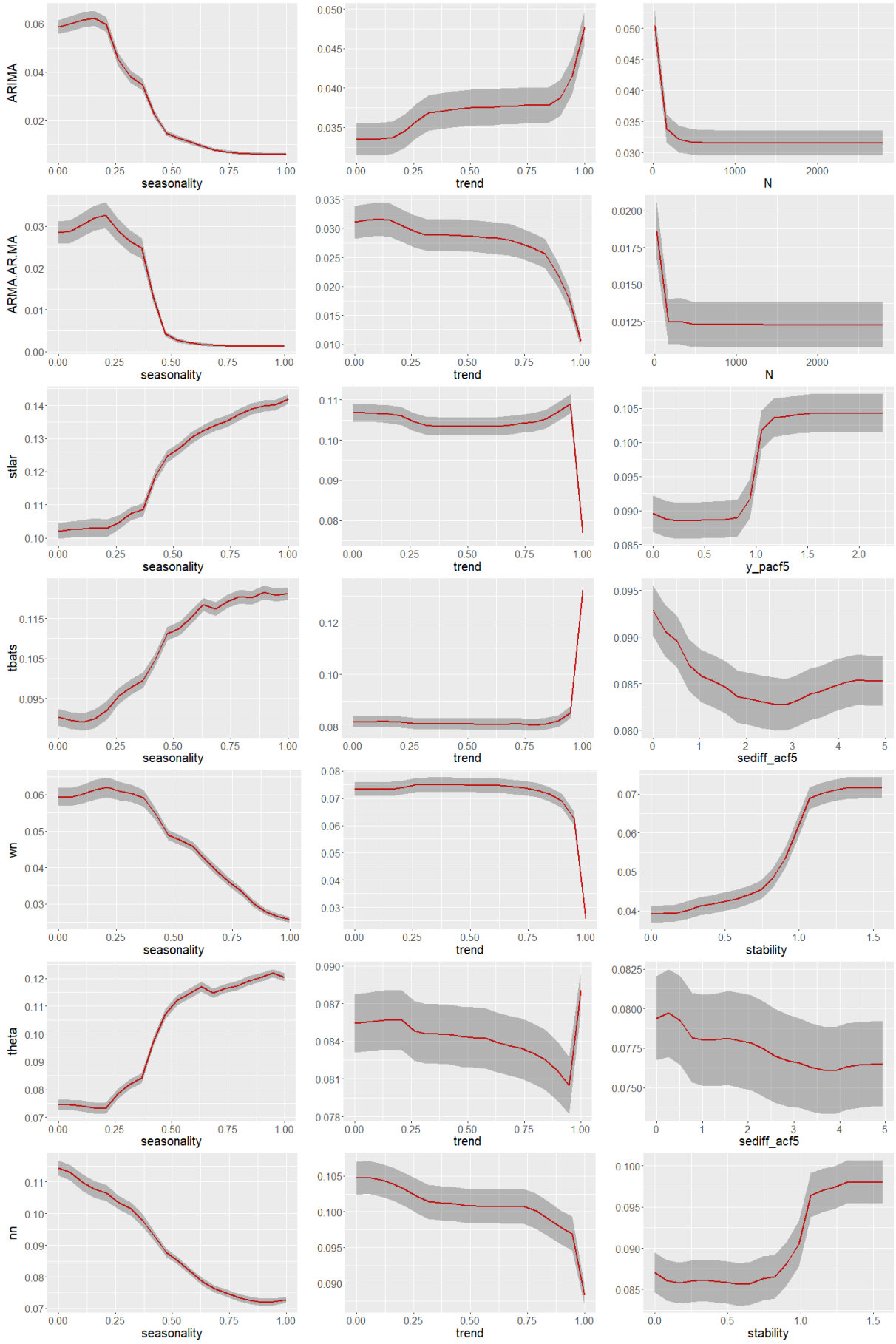
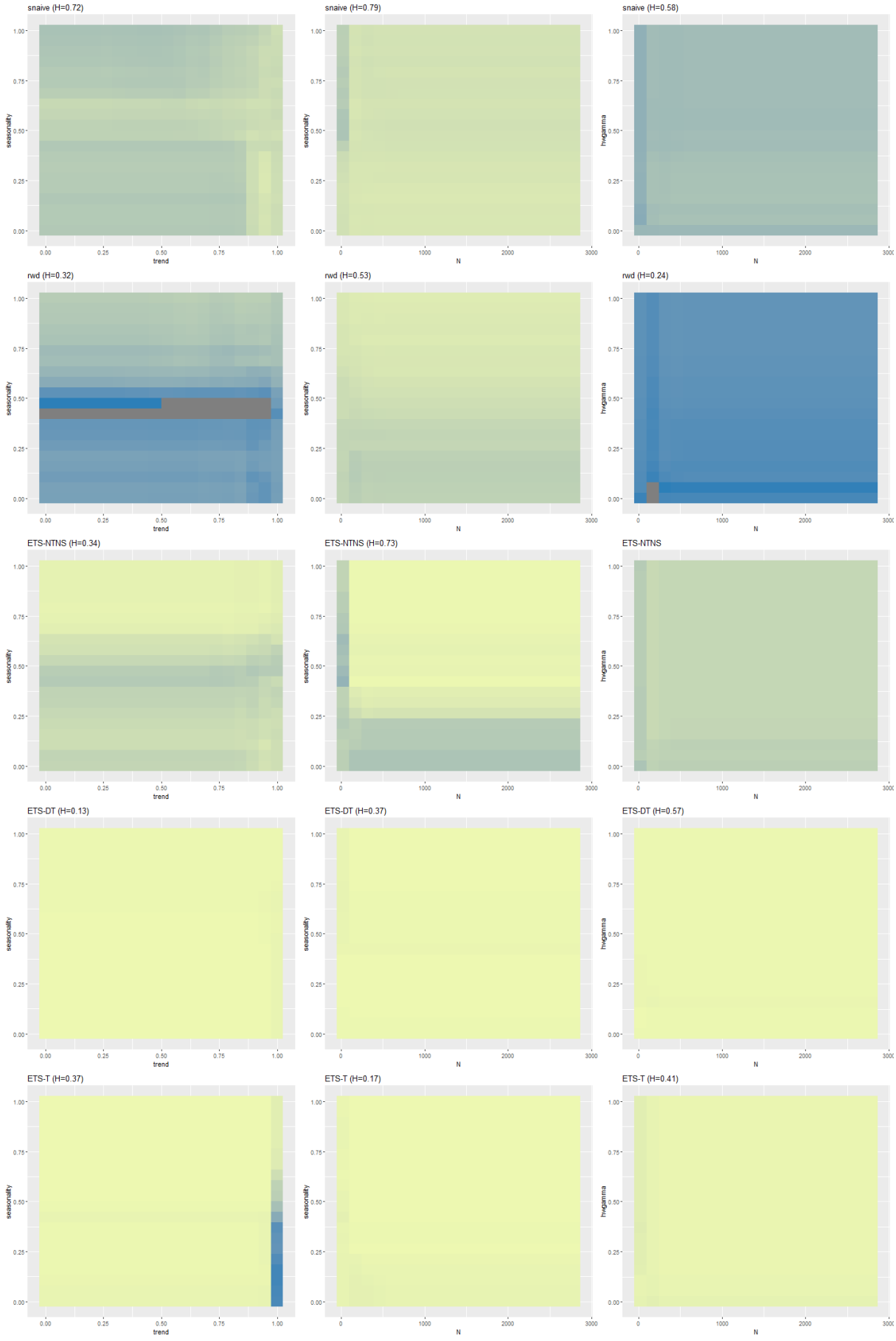


Figure 8: *(#fig:pdp_monthly2)Partial dependence plots for top three features in each category (Monthly)*


snaive (H=0.72)
snaive (H=0.79)
snaive (H=0.58)
rwd (H=0.32)
rwd (H=0.53)
rwd (H=0.24)
ETS-NTNS (H=0.34)
ETS-NTNS (H=0.73)
ETS-NTNS
ETS-DT (H=0.13)
ETS-DT (H=0.37)
ETS-DT (H=0.57)
ETS-T (H=0.37)
ETS-T (H=0.17)
ETS-T (H=0.41)
seasonality
hwgamma
trend
N

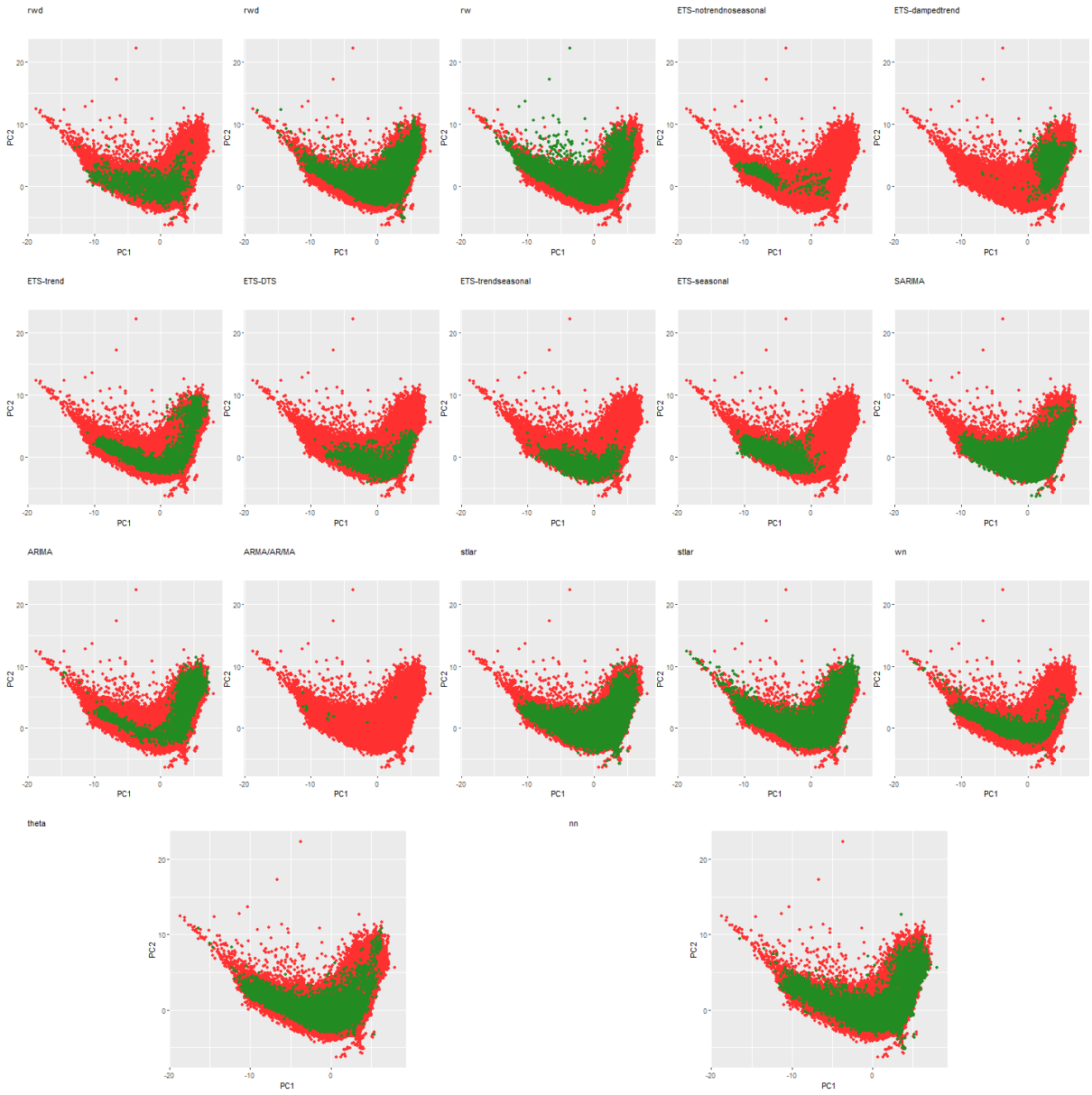


Figure 9: *(#fig:monthly_pca)PCA(Monthly)*

4.4 Hourly

5 Discussion and Conclusions

our proposed method selects a specific feature subset for each class. Feature-based time series analysis should develop support systems that incorporate these features whenever feasible and appropriate.

References

Breiman, L (2001). Random forests. *Machine Learning* **45**(1), 5–32.

Ehrlinger, J (2015). ggRandomForests: Visually Exploring a Random Forest for Regression. *arXiv preprint arXiv:1501.07196*.

Friedman, JH, BE Popescu, et al. (2008). Predictive learning via rule ensembles. *The Annals of Applied Statistics* **2**(3), 916–954.

Hyndman, R, G Athanasopoulos, C Bergmeir, G Caceres, L Chhay, M O'Hara-Wild, F Petropoulos, S Razbash, E Wang & F Yasmeen (2018). *forecast: Forecasting functions for time series and linear models*. R package version 8.3. http://pkg.robjhyndman.com/forecast.

Kang, Y, RJ Hyndman, F Li, et al. (2018). *Efficient generation of time series with diverse and controllable characteristics*. Tech. rep. Monash University, Department of Econometrics and Business Statistics.

Molnar, C, G Casalicchio & B Bischl (2018). Iml: An r package for interpretable machine learning. *The Journal of Open Source Software* **3**(786), 10–21105.

Silva, N da, D Cook & EK Lee (2017). Interactive graphics for visually diagnosing forest classifiers in R. *arXiv preprint arXiv:1704.02502*.

Sutherland, P, A Rossini, T Lumley, N Lewin-Koh, J Dickerson, Z Cox & D Cook (2000). Orca: A visualization toolkit for high-dimensional data. *Journal of Computational and Graphical Statistics* **9**(3), 509–529.

Talagala, TS, RJ Hyndman & G Athanasopoulos (2018). Meta-learning how to forecast time series. *Technical Report 6/18, Monash University.*

Zhao, Q & T Hastie (n.d.). Causal Interpretations of black-box models ().