On Gains from Biomarker Optimization

Toward ROC-Related Targets in Real-Life Data

by

Jian He

BS, University of Pittsburgh 2016

Submitted to the Graduate Faculty of the

Graduate School of Public Health in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2021

UNIVERSITY OF PITTSBURGH

GRADUATE SCHOOL OF PUBLIC HEALTH

This thesis was presented

by

Jian He

It was defended on

April 20, 2021

and approved by

Andriy I. Bandos, PhD, Associate Professor, Department of Biostatistics, Graduate School of Public Health, University of Pittsburgh

Jong H. Jeong, PhD, Professor and Interim Chair, Department of Biostatistics, Graduate School of Public Health, University of Pittsburgh

Juhun Lee, PhD, Department of Radiology, School of Medicine, University of Pittsburgh

Thesis Advisor/Dissertation Director: Andriy I. Bandos, PhD, Associate Professor, Department of Biostatistics, Graduate School of Public Health, University of Pittsburgh

On Gains from Biomarker Optimization toward ROC-Related Targets in Real-Life Data

Jian He, MS

University of Pittsburgh, 2021

Abstract

In biomedical studies, it is often of interest to classify/predict a subject's condition using a combination of multiple markers. With the introduction of additional markers, one could expect that the classification performance of a combined classification score is better than that of a single marker. However, this is not always the case. For example, the logistic regression combining two markers can be less discriminative than one of them. This phenomenon stems from the fact that logistic regression seeks to optimize a likelihood function that is not directly related to measures of classification performance. Because of these and other related problems, recent methods for marker development recommend matching the optimization targets to performance indices most relevant for the targeted application. Those optimization targets include the area under the curve (AUC), the partial AUC (pAUC) over a clinically relevant range, and the sensitivity at the lowest "tolerable" level of specificity.

In this work, I investigated and implemented several distribution-free approaches to optimizing linear combinations of prostate cancer biomarkers for a screening task, which requires high specificity of the decision rule. The primary objective is to study gains from using task-specific objective functions to optimize meaningful combinations of markers in a real-life dataset. The considered approaches range from combining markers sequentially with grid-search methods, up to combining multiple (more than 2) markers simultaneously using gradient-based optimization toward smooth approximations of classification-related objective functions.

The results indicate that combinations of real-life biomarkers can benefit substantially from optimizing the objective function tailored for the targeted classification task. The same phenomenon, possibly to a lesser degree, can be expected from less interpretable non-linear classification approaches. These findings are important in the fields of public health and medicine as a targeted optimization of biomarker combinations can substantially improve the performance of the resulting decision rules in specific tasks, such as screening a large population or triaging patients with symptoms.

Table of Contents

List of Tables

List of Figures

1.0 Introduction

In practical applications, a single marker often has limited ability to correctly classify a subject's condition (e.g., "diseased"/ "non-diseased"). Hence, it is often desirable to combine multiple markers for better discrimination accuracy. There are multiple rules for combining classification markers in the literature. We here focused on a class of rules that assign a scalar value called a "classification score" to each subject. Subjects with classification scores higher than a certain threshold are classified as potentially diseased (or "positive"), while subjects with lower scores are classified as potentially non-diseased (or "negative"). This is a rather flexible class of rules because a score can represent any mathematical combination of the input predictors. The linear combination is often used as one of the approaches to obtaining explicit and interpretable rules. Under this framework, specific marker combinations are obtained by optimizing different "objective measures/functions." Optimizing a general objective function, such as logistic likelihood (i.e., the likelihood function of the logistic regression with "diseased" as an outcome), is one of the most traditional methods for constructing marker combinations. However, over the past decade, it has been increasingly recognized that practical classification tasks could greatly benefit from using objective functions related to the receiver operating characteristic (ROC) curve. In addition, optimization toward ROC-related objective functions instead of the model-based likelihood could be beneficial when the modeling assumptions are not fully verified (Pepe et al., 2006).

For a quantitative classification score, a standard classification rule is determined by dichotomizing the results into "positive" or "negative" at a threshold ξ. Performance of a resulting classification rule is commonly characterized with sensitivity (also known as True Positive

Fraction, or $TPF(\xi)$) and specificity (which is a complement of False Positive Fraction, or $1 - FPF(\xi)$). The ROC curve, a plot of $TPF(\xi)$ against $FPF(\xi)$ for all values of threshold ξ, is a graphical device characterizing the classification performance of a quantitative marker or a classification score. A single ROC can be denoted as $TPF|_{fpf}$, with threshold ξ being implicit. The area under the ROC curve (AUC), a popular summary index of the overall classification performance, reflects the probability of correct discrimination between diseased and non-diseased subjects and ranges from 0.5 and 1 for reasonable classification scores.

Pepe et al. (2006) demonstrated that markers' combinations maximizing AUC, as opposed to logistic likelihood, could lead to a substantially more discriminative classification score. Yet, aside from specific artificial examples, it is impossible to obtain a uniformly superior ROC curve for a linear combination of markers (Anderson and Bahadur, 1962). Thus, the ROC curve of an AUC-maximizing combination can be locally lower than the ROC curve for another linear combination. More specifically, combining markers to maximize AUC can result in suboptimal characteristics for specific practical applications, such as the task of screening a large population for rare diseases.

In the example mentioned, limited resources, combined with the intent to spare healthy people of unnecessary procedures, drive the requirement for high specificity (e.g., the national benchmark for the abnormal interpretation rate in screening mammography is approximately 10%, Lehman et al., 2017). Thus, a clinically relevant combination of markers would aim at improving sensitivity levels of the resulting classification score at the thresholds that lead to high specificity (e.g., 90% or higher). A straightforward approach to constructing relevant classification scores is using part of the ROC curve over the high-specificity region (i.e., $\xi{:}FPF(\xi) \leq 0.1$) as an objective function for optimization (e.g., Wang and Chang, 2011; Komori and Eguchi, 2010). Due to the

monotonicity of the ROC curve, one of the most natural objective functions is sensitivity at the minimum tolerable specificity, or $TPF|_{fpf}$ (e.g., Meisner et al., 2017). For example, marker combinations that maximize $TPF|_{fpf=0.1}$ are designed to achieve the maximum sensitivity while constraining specificity to the $> 90\%$ range.

The gains in sensitivity achieved by using a task-specific objective function, $TPF|_{fpf=0.1}$, instead of a global classification-oriented AUC, for combining markers can be substantial (Bandos and Gur, 2017). This can be illustrated with a simple theoretical example of two conditionally independent markers T_1 and T_2 ($T_1 \perp T_2|D$, where $D = 0,1$, indicates a disease status), with normally distributed values for diseased and non-diseased subpopulations (i.e., $T_i\ |D \sim N(\mu_{D_i}, \sigma^2_{D_i})$). Figure 1 shows the ROC curves for classification scores based on the linear combinations of such markers (i.e., for U=β`T) where different β's were selected to maximize $TPF|_{fpf=0.1}$ or AUC of the resulting classification score U. Without loss of generality, values of both markers for the non-diseased subpopulation were modeled by a standard normal distribution (i.e., s. For the diseased subpopulation, values of the markers followed the normal distribution with the parameters leading to AUCs of 0.65 and 0.75, namely, $T_1|D = 1 \sim N(1.96,25)$ and $T_2|D = 1 \sim N(0.69,0.04)$, respectively. The coefficients of the linear combination maximizing AUC (i.e., β_{AUC}) were computed using the available exact solution (Su and Liu, 1993). The coefficients of linear combination maximizing $TPF|_{fpf=0.1}$ (i.e., β_{TPF}) were found using grid search based on the known distribution parameters. The ROC curves and related summary indices for the resulting combinations were determined by the closed-form solutions for the "binormal" markers (Zhou et al., 2011).

The ROC curves in Figure 1 demonstrate that by directly targeting $TPF|_{fpf=0.1}$ the given markers can be combined to achieve $TPF|_{fpf=0.1} = 0.57$, whereas the combination maximizing

AUC offers substantially lower level of sensitivity in the targeted range of specificity ($TPF|_{fpf=0.1} = 0.26$). The coefficients of the two combinations are substantially different, i.e. ($\beta_{TPF} = (0.84,0.54)$ versus $\beta_{AUC} = (0.11,0.99)$, indicating a large difference in the needs of the two objective functions even in this simple example. The gains and differences in the real-life data could be even more substantial. However, in contrast to the theoretical example, the corresponding investigation, which primarily centers around studying the gains from using the TPF-based objective function in the real-life data of finite size, are complicated by the need to account for sampling variability and related phenomena.

The remainder of this work is organized as follows. In Section 3, we introduce procedures for optimizing combinations of markers toward the two task-specific objective functions, namely AUC and $TPF|_{fpf}$. The results from optimization toward different objective functions in the real-life dataset are compared with each other and with the results from lasso, ridge regression, and random forests in Section 4. We summarize and discuss the key findings in Section 5. The next section focuses on the description of the prostate cancer dataset and the presentation of the results from a descriptive analysis of the data.

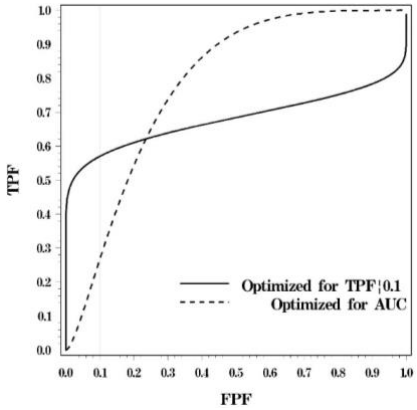

Figure 1 The ROC curves for the theoretical marker combinations $\boldsymbol{\beta`T}$ optimizing the true $\boldsymbol{TPF|_{fpf=0.1}}$ ($\boldsymbol{\beta = (0.84, 0.54)}$, AUC=0.68, $\boldsymbol{TPF|_{fpf=0.1} = 0.57}$) and optimzing the true AUC ($\boldsymbol{\beta = (0.11, 0.99)}$, AUC=0.78, $\boldsymbol{TPF|_{pf=0.1} = 0.26}$).

2.0 Dataset: Prostate Cancer Biomarkers

We explored the possible real-life gains from optimizing marker combinations toward screening-oriented performance (i.e., maximum sensitivity for high-specificity decisions) with examples of biomarkers for prostate cancer. We used data on quantitative biomarkers obtained in the protein mass spectrometry study (Yasui, et al., 2003), which had been used in a work on combining markers to maximize the area under the ROC curve, or AUC (Pepe et al., 2006; FHCR, DABS/datasets). The dataset contains values of 15 pre-processed protein biomarkers for 167 serum samples of different patients with verified prostate cancer and 81 men without cancer (Pepe et al., 2006). We note that this is an illustrative dataset that contains a selected set of biomarkers, not all of which are necessarily important for distinguishing between cancer and non-cancer patients. This dataset, however, presents a wide spectrum of biomarkers' classification characteristics that can be encountered in practice.

Table 1 summarizes the basic classification-related characteristics of each of 15 biomarkers, with individual ROC curves for detecting cancer samples illustrated in Figure 2. Four of fifteen biomarkers (i.e., v30, v182, v354, and v365) had a smaller median value in cancer patients than in non-cancer patients (and would have resulted in empirical AUCs < 0.5). Without loss of generality for the current investigation of marker combinations, low values of these markers were used to indicate the presence of cancer (which results in all empirical AUCs > 0.5). For the other biomarkers, high values were considered indicative of the presence of cancer.

Four of fifteen biomarkers (v30, v182, v365, and v426) were not univariately significant for discriminating between cancer and non-cancer patients (AUC from 0.51 to 0.56, with all p-values > 0.09). One biomarker, v427, with a fair discrimination ability (AUC = 0.58, with 95%

CI: 0.51-0.65) was not statistically significant within the framework of evaluating fifteen distinct biomarkers ($p = 0.03$). Other ten biomarkers (v93, v354, v509, and those with higher numbers) had at least moderate and statistically significant ability to discriminate between serum samples with and without prostate cancer (AUC from 0.63 to 0.73, all p-values < 0.001). We note, however, regardless of the univariate significance any of the fifteen biomarkers can be significant contributors to the performance of a combination of multiple biomarkers (Bansal and Pepe, 2013). Thus, all fifteen biomarkers would be considered in further investigation of multi-marker combinations.

At the initial stages of constructing multi-marker combinations, individual biomarkers are often ordered by the level of their individual performance. A standard approach in statistics is to order biomarkers by the p-values from logistic regression (with prostate cancer as an outcome). Whereas a general classification-oriented approach is to order biomarkers according to the p-values of the test for the null hypothesis that AUC is equal to 0.5 (e.g., based upon Delong et al., 1988). In our dataset, biomarkers significant under the logistic regression formed a subset of non-trivial biomarkers. For biomarkers v93, v354, and v530, the p-values from the logistic regression were 0.45, 0.73, and 0.37, respectively, whereas all corresponding AUC-based p-values were less than 0.001). This observation echoes the phenomenon illustrated by Pepe et al. (2006) and highlights the importance of using classification-oriented measures to identify and combine biomarkers.

Testing for statistical significance of AUC is theoretically sufficient to identify non-trivial markers (Pepe et al., 2013). However, this approach is not uniformly most powerful, and the relative importance of individual markers for some specific classification tasks could differ from their discrimination ability. For example, among ten individually non-trivial biomarkers, the most

promising for screening was the marker v354 (AUC = 0.68; 95 % CI: 0.61-0.74) with empirical $TPF|_{fpf=0.1} = 0.44$, whereas the most discriminative marker v831 (AUC = 0.73; 95 % CI: 0.66-0.79) had empirical $TPF|_{fpf=0.1} = 0.40$. (We note, however, that for the current dataset there was no statistically significant difference in classification accuracy of biomarkers v354 and v831, p = 0.12 for the difference in AUCs.) Discrepancies in the relative importance of individual biomarkers for the overall discrimination and screening tasks, indicate a high potential for substantial differences in the composition and weights of combinations maximizing $TPF|_{fpf}$ versus AUC. As shown in Figure 2, the ROC curves of the considered prostate cancer biomarkers span a wide spectrum of shapes, including curves that correspond to theoretical scenarios where a large gain from the targeted optimization can be expected (e.g., Figure 1).

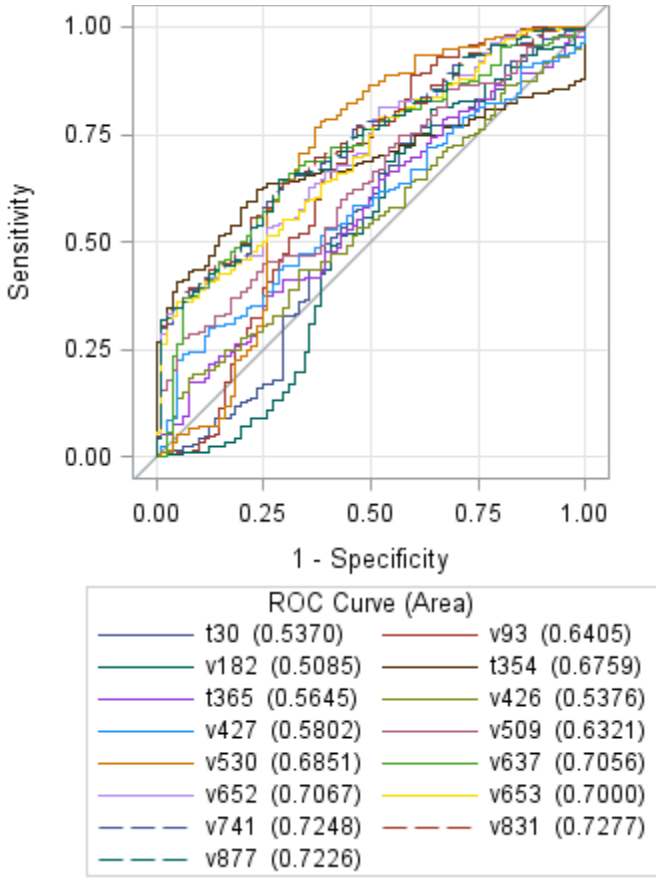

Figure 2 The ROC curve for each biomarkers in the prostate cancer dataset.

Table 1 Estimated characteristics of individual biomarkers (for 167 cancer and 81 non-cancer serum samples).

Biomarkers*	Median value		p-value	AUC	$TPF\|_{fpf=0.1}$	pAUC(0,0.1)
	non-cancer	cancer	(logistic regression)	(†p-value)	(‡p-value)	(§p-value)
v182**	-0.07	-0.1	0.548	0.51 (0.846)	0.012 (<0.001)	0.001 (<0.001)
v30*	0.47	0.44	0.697	0.54 (0.379)	0.042 (0.073)	0.002 (0.0298)
v426	-0.15	-0.14	0.777	0.54 (0.326)	0.192 (0.045)	0.01 (0.267)
v365*	-0.08	-0.12	0.382	0.56 (0.095)	0.174 (0.172)	0.01 (0.232)
v427	0.24	0.26	0.246	0.58 (0.031)	0.246 (0.003)	0.015 (0.079)
v509	-0.31	-0.29	<0.001	0.63 (<0.001)	0.293 (<0.001)	0.021 (0.001)
v93	0.19	0.29	0.449	0.64 (<0.001)	0.036 (0.304)	0.001 (0.009)
v354* (best TPF)	**0.48**	**0.29**	**0.725**	**0.68 (<0.001)**	**0.437 (<0.001)**	**0.038** (<0.001)
v530	-0.39	-0.31	0.371	0.69 (<0.001)	0.072 (0.443)	0.004 (0.567)
v653	-0.46	-0.42	<0.001	0.7 (<0.001)	0.389 (<0.001)	0.03 (<0.001)
v652	-0.46	-0.42	<0.001	0.71 (<0.001)	0.389 (<0.001)	0.031 (<0.001)
v637	-0.46	-0.41	0.033	0.71 (<0.001)	0.395 (<0.001)	0.021 (0.045)
v741	-0.48	-0.43	<0.001	0.72 (<0.001)	0.395 (<0.001)	0.031 (<0.001)
v877	-0.48	-0.43	<0.001	0.72 (<0.001)	0.401 (<0.001)	0.031 (<0.001)
v831(best AUC)	**-0.48**	**-0.43**	**<0.001**	**0.73 (<0.001)**	**0.401 (<0.001)**	**0.032 (<0.001)**

* *presented in an ascending order by the empirical AUC.*
† *Non-parametric asymptotic test for H_0: AUC=0.5.*
‡ *Non-parametric asymptotic test for H_0: $TPF|_{fpf=0.1}=0.1$.*
§ *Non-parametric asymptotic test for H_0: pAUC(0,0.1)=0.005.*
** *Biomarkers with low values (empirically) more indicative for cancer. These are transformed for computing AUC.*

3.0 Methodology

3.1.1 Biomarker Combinations Maximizing Empirical Objective Functions

We assume that p biomarkers $\boldsymbol{X} = (\boldsymbol{X_1}, \ldots, \boldsymbol{X_p})$ are available for every subject. For notational convenience, we use $\boldsymbol{X}_{1i}$ to denote the vector of values of p markers for the i^{th} diseased subject ($i = 1, \ldots, n_1$) and $\boldsymbol{X}_{0j}$ denote the vector of values of the same p markers for the j^{th} non-diseased subject ($j = 1, \ldots, n_0$).

We are interested in estimating the coefficients of the linear combination of biomarkers, $\boldsymbol{U} = \beta^T \boldsymbol{X}$, that maximize a given objective function. For combination maximizing AUC, we are searching for

$$\beta_0 = \arg\max_{\beta \in \mathbf{B}} \; AUC(\beta),$$

where $\mathbf{B} = \{\beta \in \mathcal{R}^p : \|\beta\| = 1\}$ and $AUC = P(\beta^T \boldsymbol{X}_1 > \beta^T \boldsymbol{X}_0)$. The restriction of the vectors of combination coefficients to a sphere resolves the identifiability problem with maximizing ROC-related targets (since the ROC curve is invariant with respect to monotone transformations of the classification score).

For practical purposes, it is natural to replace the unknown true AUC with its empirical estimate (e.g., Pepe et al., 2006), which for continuous biomarkers can be formulated as follows:

$$\widehat{AUC} = \sum_{j=1}^{n_0} \sum_{i=1}^{n_1} I(\beta^T \boldsymbol{X}_{1i} > \beta^T \boldsymbol{X}_{0j}). \qquad (1)$$

Equation (1) then leads to the following formulation of the estimate for the combination coefficients,

$$\hat{\beta} = \arg\max_{\beta \in \mathbf{B}} \frac{1}{n_0 n_1} \sum_{j=1}^{n_0} \sum_{i=1}^{n_1} I(\beta^T \boldsymbol{X}_{1i} > \beta^T \boldsymbol{X}_{0j}). \quad (2)$$

The estimation of the combination coefficients is straightforward when only a few biomarkers are being considered. For combining two biomarkers, a nonparametric estimate of β_0 can be obtained by conducting a single-parameter grid search over a bounded set of the polar angle $\gamma \in (0,360°]$, which defines the vector of coefficients as $\beta_1 = cos(\gamma\pi/180°)$ and $\beta_2 = sin(\gamma\pi/180°)$. For combining multiple (>2) biomarkers, either the sequential incorporation can be done using a grid search, or multiple biomarkers can be combined simultaneously by using gradient-based methods or other multivariable techniques. The latter, however, requires the use of smooth object functions (e.g., smooth approximations to the empirical ROC indices, which will be described in Section 3.1.2).

Once the combination coefficients are estimated, the corresponding classification score, $\boldsymbol{U} = \beta^T \boldsymbol{X}$, can be constructed and evaluated. For example, we can estimate the classification score's overall discrimination ability with $\widehat{AUC}(\hat{\beta})$ obtained according to equation (1). However, such an estimate would be too optimistic because using the same data for training and testing would result in a positive bias ("optimism"). The unbiased estimate of classification performance can be obtained from validation data (testing set) which are independent of the data used for estimating the marker combination (training set). Splitting data into fixed training and testing sets is, however, often impractical for datasets with small sample sizes.

Cross-validation can be used to obtain a less biased estimate for $AUC(\hat{\beta})$ from the same dataset. The standard K-fold cross-validation procedure splits data $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_0)$ into k parts. (folds) $\{\boldsymbol{X}^f = (\boldsymbol{X}_0^f, \boldsymbol{X}_1^f)\}_{f=1}^k$, and uses the f's fold, $\boldsymbol{X}^f$, as the testing set for the combination

coefficients estimated on the rest of the data ($\boldsymbol{X}^{(-f)}$). The most standard cross-validated estimate of AUC can then be formulated as follows:

$$AUC^{(CV)} = \frac{1}{n_0 n_1} \sum_{f=1}^{k} \sum_{j} \sum_{i} I(\hat{\beta}^{(-f)T} \boldsymbol{X}_{1i}^{f} > \hat{\beta}^{(-f)T} \boldsymbol{X}_{0j}^{f}), \tag{3}$$

where i and j enumerate subjects within fold f, and $\hat{\beta}^{(-f)}$ is derived according to equation (2) on data without the f^{th} fold. The estimator in (3) pools the fold-specific data to compute a single empirical AUC. An alternative approach is to compute the fold-specific empirical AUCs and use their average as an overall cross-validated estimate. The first "pooled" estimator is often associated with a negative bias (i.e., overcompensates the bias of re-substitution estimates), while the average estimate has substantial variability (Airola, et al., 2011), especially when individual folds have small size. We will focus on the pooled cross-validation estimation due to its higher precision, while using the average estimator for verifying the magnitude of the pooling-related bias.

The same general approach can be extended to estimate the biomarker combination that maximize other ROC-related objective functions, such as $TPF|_{fpf}$, which is an ROC index especially relevant for evaluating classification scores designed for screening a large population.

For any fixed $t \in (0,1)$, we consider

$$(\beta_t, \xi_t) = \underset{(\beta,\xi)\in\boldsymbol{\Omega_t}}{\arg max}\ TPF(\beta, \xi),$$

where $\Omega_t = \{\beta \in \mathcal{R}^p, \xi \in \mathcal{R}: \|\beta\| = 1, FPF(\beta, \xi) \leq t\}$. For practical implementation, we replace the unknown TPF and FPF characteristics by their empirical estimates,

$$\widehat{TPF}(\beta, \xi) = \frac{1}{n_1} \sum_{i=1}^{n_1} I(\beta^T \boldsymbol{X}_{1i} > \xi), \widehat{FPF}(\beta, \xi) = \frac{1}{n_0} \sum_{J=1}^{n_0} I\left(\beta^T \boldsymbol{X}_{0j} > \xi\right).$$

and then define

$$(\widehat{\beta_t}, \hat{\xi}_t) = \underset{(\beta,\xi)\in\widehat{\boldsymbol{\Omega}}_{t,n_0}}{\arg max}\ \widehat{TPF}(\beta, \xi), \tag{4}$$

where $\widehat{\boldsymbol{\Omega}}_{t,n_0} = \{\beta \in \mathcal{R}^p, \xi \in \mathcal{R}: \|\beta\| = 1, \widehat{FPF}(\beta, \xi) \leq t\}$.

3.1.2 Smooth Approximations to the Empirical ROC Indices

So far, we have focused our discussion on optimizing linear combinations of two markers toward the empirical ROC indices using a simple grid search, which does not require a smooth objective function. Since the simultaneous grid search for a combination of more than two or three markers is computationally difficult, a sequential approach by adding one marker at a time may be considered (Pepe and Thompson, 2000). However, as a greedy algorithm it can often result in suboptimal global solutions. This problem is commonly addressed by developing a smooth approximation to the desired empirical objective function and applying a simultaneous gradient-based search for optimal combination coefficients.

For the AUC-based optimization of biomarker combinations, Fong et al., 2016, proposed the smoothed AUC (SAUC) method that is based upon a smooth approximation to the indicator function in equation (1), namely $I(w > 0) \approx \Phi(w/s)$, where Φ is the cumulative distribution function (or CDF) of the standard normal distribution and s is a tuning parameter that controls the level of smoothing (Lin, et al., 2011). The corresponding vector of optimal combination coefficients can be formulated accordingly:

$$\tilde{\beta} = \arg\max_{\beta \in \mathbf{B}} \frac{1}{n_0 n_1} \sum_{j=1}^{n_0} \sum_{i=1}^{n_1} \Phi\left(\beta^T (\boldsymbol{X}_{1i} - \boldsymbol{X}_{0j})/s\right), \tag{5}$$

where $\mathbf{B} = \{\beta \in \mathcal{R}^p: \|\beta\| = 1\}$. Certain considerations, including the choices of the tuning parameter s and the starting value $\tilde{\beta}$ for procedure (5), need to be addressed before the

implementation of any gradient-based optimization. $s = \tilde{\sigma} n^{-1/2}$ is suggested by Meisner et al. (2017), where $\tilde{\sigma}$ is the sample standard error of $\beta^T \boldsymbol{X}$ and n is the total sample size.

A similar procedure can also be applied to the indicator functions involved in equation (4) to construct the smooth approximations to the empirical TPF and FPF as follows:

$$\widetilde{TPF}(\beta, \xi) = \frac{1}{n_1} \sum_{i=1}^{n_1} \Phi\left(\frac{\beta^T \boldsymbol{X}_{1i} - \xi}{s}\right), \widetilde{FPF}(\beta, \xi) = \frac{1}{n_0} \sum_{J=1}^{n_0} \Phi\left(\frac{\beta^T \boldsymbol{X}_{0j} - \xi}{s}\right).$$

The above smooth approximations allow the use of gradient-based optimization, which can then be used to compute

$$\left(\tilde{\beta}_t, \tilde{\xi}_t\right) = \underset{(\beta,\xi) \in \widetilde{\boldsymbol{\Omega}}_{t,n_0}}{\arg max} \ \widetilde{TPR}_{n_1}(\beta, \xi),$$

where $\widetilde{\boldsymbol{\Omega}}_{t,n_0} = \left\{\beta \in \mathcal{R}^p, \xi \in \mathcal{R}: \|\beta\| = 1, , \widetilde{FPF}(\beta, \xi) \leq t\right\}$. These steps outline the multi-layer approximation which is necessary for the implementation of gradient-based algorithms to achieve simultaneous optimization. The complexity of the approximation might affect the performance of the resulting estimates, especially for small sample sizes. Thus, in the conducted investigation of multi-marker combinations, we consider the estimates obtained by both grid search and gradient-based optimization.

3.1.3 Considered Methods for Combining Multiple Markers

A part of this work centers around the performance comparison across several standard approaches to optimizing the linear combinations of multiple (up to all fifteen) biomarkers in the dataset. We categorize the considered approaches for such an optimization problem into two general classes of methods: the **sequential** incorporation of biomarkers and the **simultaneous** optimization of all 15 biomarkers. The sequential approaches allow the use of the grid search method, which can deal with non-smooth objective functions but gives rise to suboptimal solutions

more often than the gradient-based method does. The simultaneous approaches, on the other hand, typically require differentiable objective functions, thereby necessitating the use of smooth approximations to ROC indices in our investigation, which might lead to suboptimal performance of the resulting classification score in terms of ROC measures.

Within the class of sequential approaches, we consider sequentially adding individual biomarkers to the existing linear combination based on p-values from logistic regression, empirical AUC, and empirical $TPF|_{fpf}$. This procedure of forward selection does not stop until the p-value is less than 0.05 or there is no improvement in the corresponding ROC measures. Within the class of simultaneous approaches, we consider likewise the general likelihood-based methods as well as classification-oriented ones. Lasso and ridge regression are regression methods that perform both variable selection and regularization to enhance classification performance and interpretability of the classifiers they produce. SAUC and maxTPR estimate linear combinations of biomarkers by maximizing the smooth approximations to the empirical estimates of AUC and $TPF|_{fpf}$ (described in Section 3.1.1). Lastly, the result of random forests (R package "RandomForest", v.4.6-13; Liaw and Weiner, 2002), which combine all fifteen biomarkers in a non-linear manner, is used as a benchmark of the achievable classification performance by an optimized classifier.

4.0 Application to the Dataset of Prostate Cancer Biomarkers

We used the dataset of prostate cancer biomarkers (described in Section 2) to investigate possible real-life gains in classification performance from maximizing general and classification-related objective functions. We first analyzed combinations of 105 pairs of biomarkers optimized toward logistic regression, AUC, and $TPF|_{fpf}$. The maximum likelihood approach was used for optimizing logistic likelihood, while the grid search method was applied to optimization toward empirical AUC and $TPF|_{fpf}$ and their smooth approximations. We then extended our investigation into combinations of multiple biomarkers, where we used the grid search method to sequentially construct the combinations maximizing the empirical ROC indices as well as the gradient-based method to simultaneously determine the linear combinations of all 15 biomarkers maximizing the smooth approximations to the corresponding ROC indices. Finally, we used a method of random forests to combine all 15 biomarkers non-linearly to access the extent of depreciation in classification performance resulting from more interpretable linear combinations.

4.1.1 Combinations of Two Biomarkers

4.1.1.1 Maximizing AUC versus Logistic Likelihood

Due to the lack of a perfectly correct model, optimization of a linear combination of biomarkers toward AUC instead of likelihood can lead to a more accurate classification score (Pepe et al., 2006). We analyzed the frequency and magnitude of non-trivial improvements in the real data using pairwise linear combinations of 15 biomarkers for prostate cancer (105 pairs overall), optimized using a simple grid search and the maximum likelihood approach for AUC and

logistic likelihood, respectively. Figure 3 demonstrates that, as expected in training data, the biomarkers' combinations that maximize AUC always achieve better classification performance in terms of empirical AUCs (EAUCs) than those that maximize the logistic likelihood.

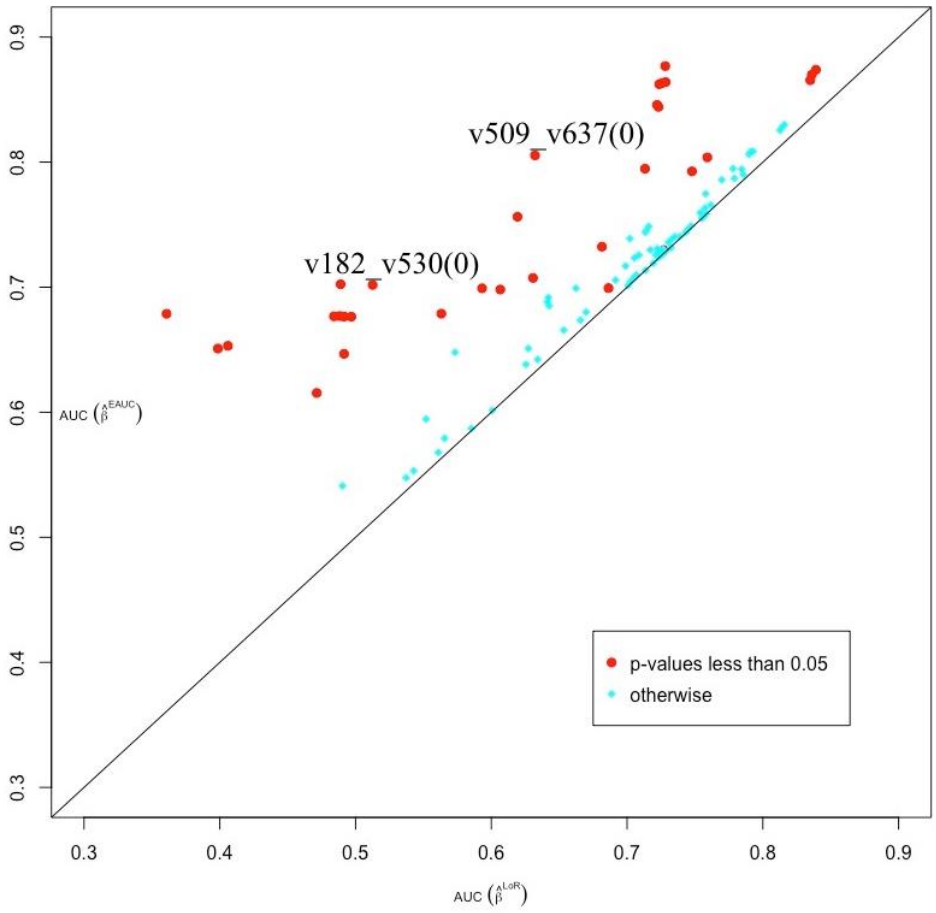

Figure 3 The training EAUCs of the biomarker pairwise combinations maximizing the EAUC versus logistic likelihood.

Many of the resulting differences in EAUCs were nontrivial and could be substantial (e.g., the biomarker pairs, v182&v530, v509&v637). In each of the two examples shown in Figure 4, comparing to logistic likelihood, maximizing AUC directly led to gains in EAUC of 0.19 and **0.18**, respectively. We also note that, in both examples, maximizing the logistic likelihood yielded a classification score with poorer discrimination ability than that of a single component marker (EAUCs: 0.51 for v182&v530 and 0.63 for v509&v637 versus 0.69 for v530 and 0.71 for v637).

The same pairs of biomarkers had been discussed in the work by Pepe et al. (2006). Many other pairs with substantial gains from optimization toward AUC could be identified. The current work does not aim to identify all instances with substantial gains from optimizing toward AUC instead of logistic likelihood, but rather to explore the major trends and notable features of biomarkers' performance that can benefit from the targeted optimization. Figure 3 illustrates that the largest gains from optimizing toward AUC occurred when the empirical AUC from the logistic-regression score was close to 0.5, however, quite substantial gains persisted for AUCs larger than 0.7. The examples shown in Figure 4 demonstrate that substantial gains occurred for biomarkers with irregularly shaped ROC curves (e.g., partially below the diagonal) as well as for biomarkers with nearly concave ROC curves.

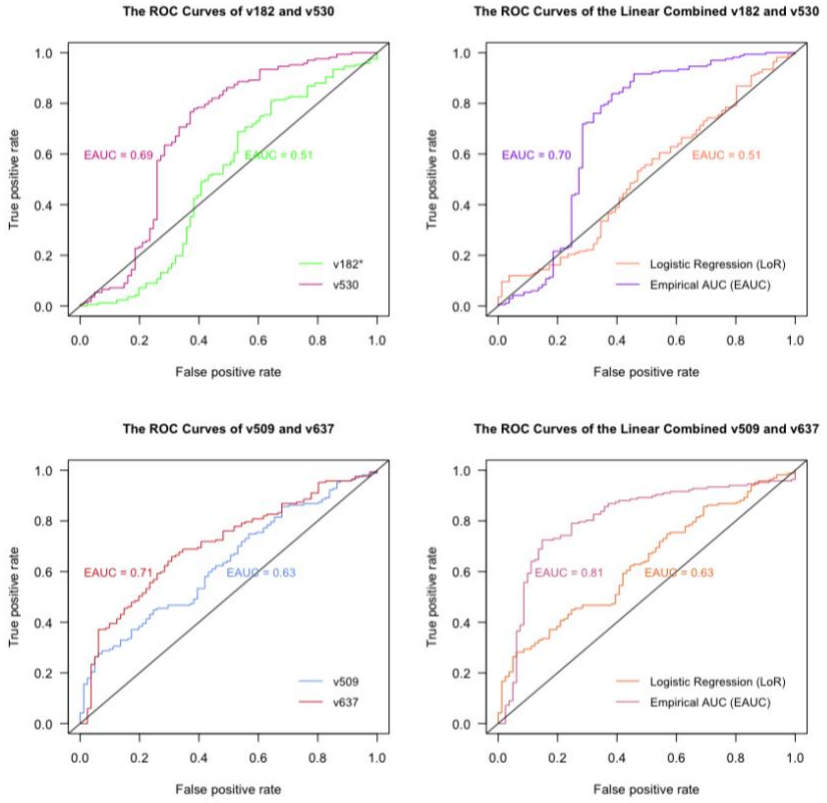

Figure 4 The training ROC curves of the combinations of the selected biomarker pairs maximize the EAUC versus logistic likelihood.

** Top left plot shows the ROC curve for the reversed values of biomarker v182.*

The gains demonstrated in Figure 3 were, however, somewhat optimistic, as the same data was used for estimating the biomarker combinations (i.e., "training") and estimating their classification performance (i.e., "testing"). To adjust for the optimism of these re-substitution estimates, we implemented a 10-fold cross-validation procedure described in Section 3. Figure 5 shows the cross-validated estimates of EAUC from logistic regression and direct optimization (toward AUC; thereby representing the bias-adjusted version of Figure 3). In contrast to Figure 3, some points in Figure 5 lay below the diagonal indicating that in rare instances logistic regression led to marker combinations that had higher testing AUC than those maximizing the empirical AUC in the training sample. However, most points were above the diagonal, suggesting prevailing gains from the AUC optimization.

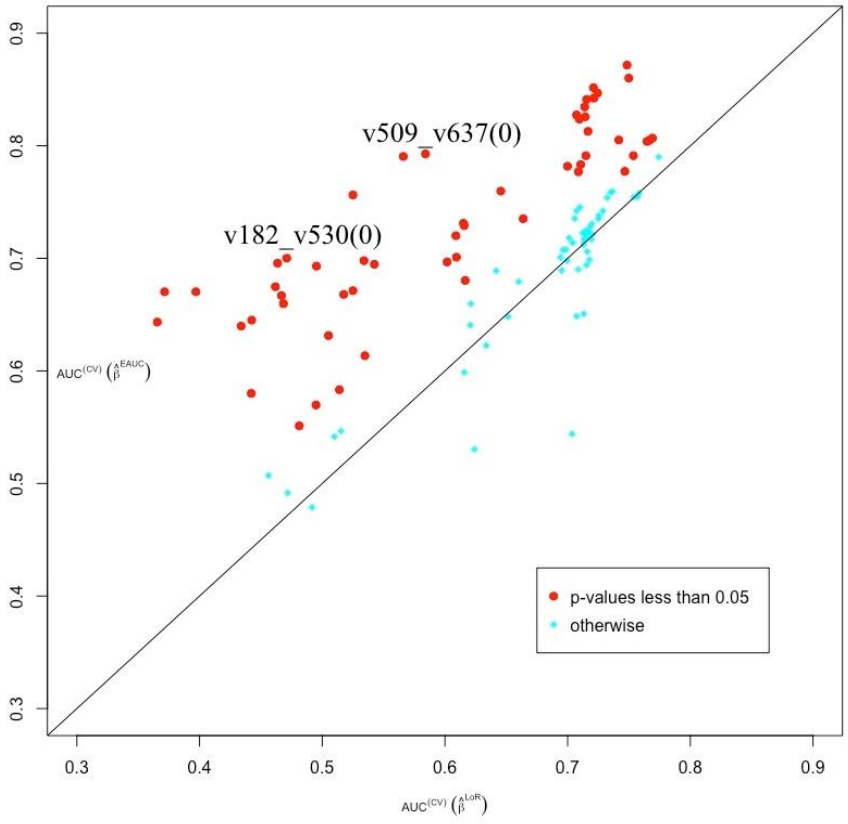

Figure 5 The cross-validated EAUCs of the biomarker pairwise combinations maximizing the EAUC versus logistic likelihood.

Interestingly, points in Figure 5 were overall located higher above the diagonal than those in Figure 3. For example, for a pair of biomarkers, v509&v637, the decrease due to cross-validation was larger for the combination maximizing the logistic likelihood ($0.05 = 0.63 - 0.58$) than for those maximizing the EAUC ($0.02 = 0.81 - 0.79$), thereby resulting in a larger estimate of the gain (i.e., the difference in EAUCs of 0.21, $p < 0.001$).

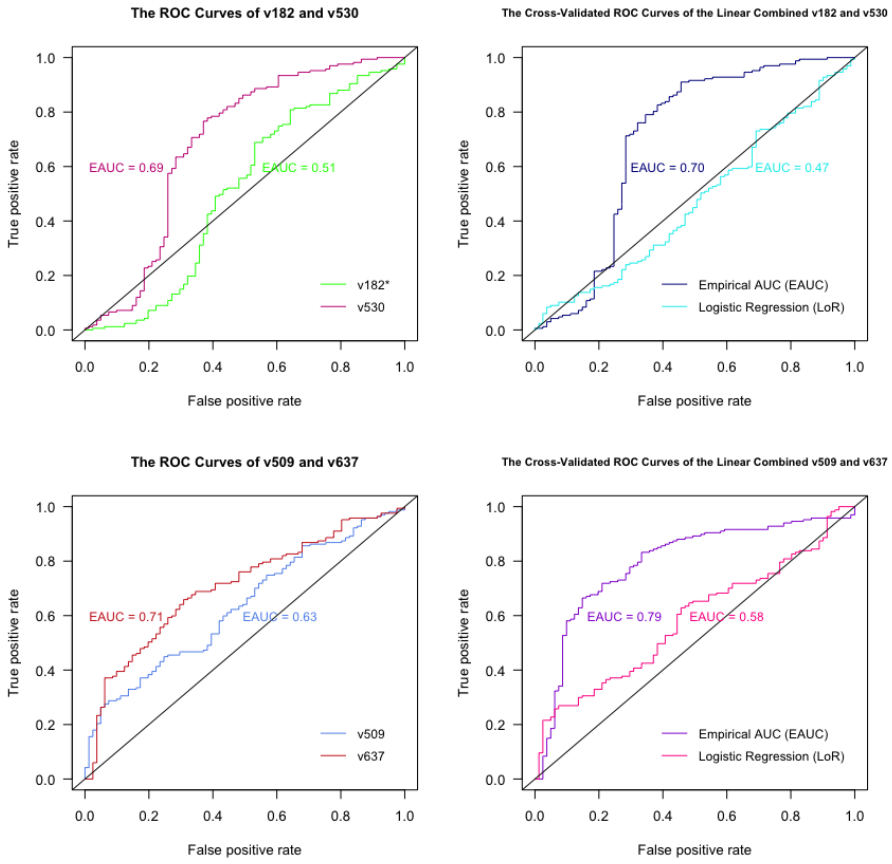

Figure 6 The cross-validated ROC curves of the combinations of the selected biomarker pairs maximizing the EAUC versus logistic likelihood.

** Top left plot shows the ROC curve for the reversed values of biomarker v182.*

As evident from Figures 4 and 5, the shapes of the cross-validated ROC curves for the combinations of the selected biomarker pairs remained like the training estimates, even though their numeric levels demonstrated larger gains from the targeted optimization. For both pairs, the classification scores based on the logistic regression resulted in smaller AUCs than that for a single

biomarker (0.47 for the combination versus 0.69 for v530 alone, $p < 0.001$, and 0.58 for the combination versus 0.71 for v637 alone, $p = 0.006$).

4.1.1.2 Maximizing a Smooth Approximation to AUC

For later assessments of the gradient-based method, we also examined the performance of pairwise biomarker combinations that maximize a smooth approximation to AUC (described in Section 3.1.1). We compared the results for the objective function in the form of a smooth approximation to AUC (SAUC) with those for EAUC (which we previously considered in Figures 3, 4, 5, and 6). Figure 7 demonstrates that there was no substantial difference between the training EAUCs achieved by the two methods across the 105 pairs of biomarkers. Figure 8 illustrates the training ROC curves for the combinations of the previously considered pairs of biomarkers, v182&v530 and v509&v637, optimized toward SAUC and EAUC. In both cases, the ROC curves are almost identical.

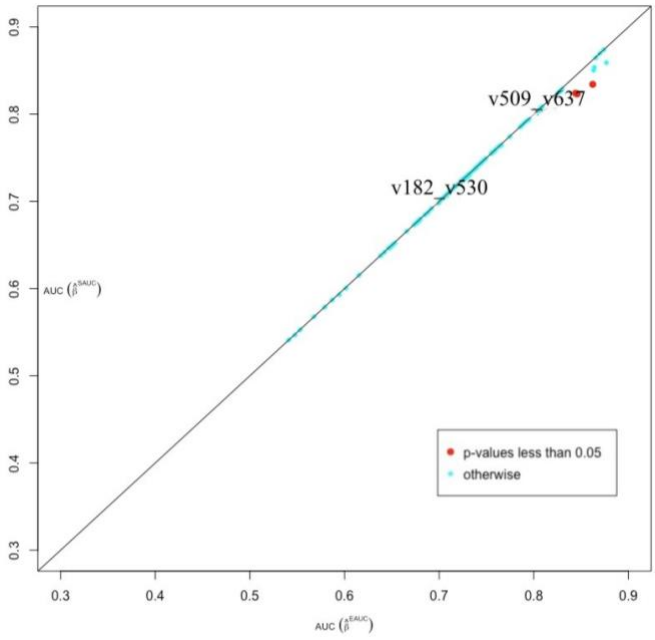

Figure 7 The training EAUCs of the biomarker pairwise combinations maximizing the SAUC versus EAUC.

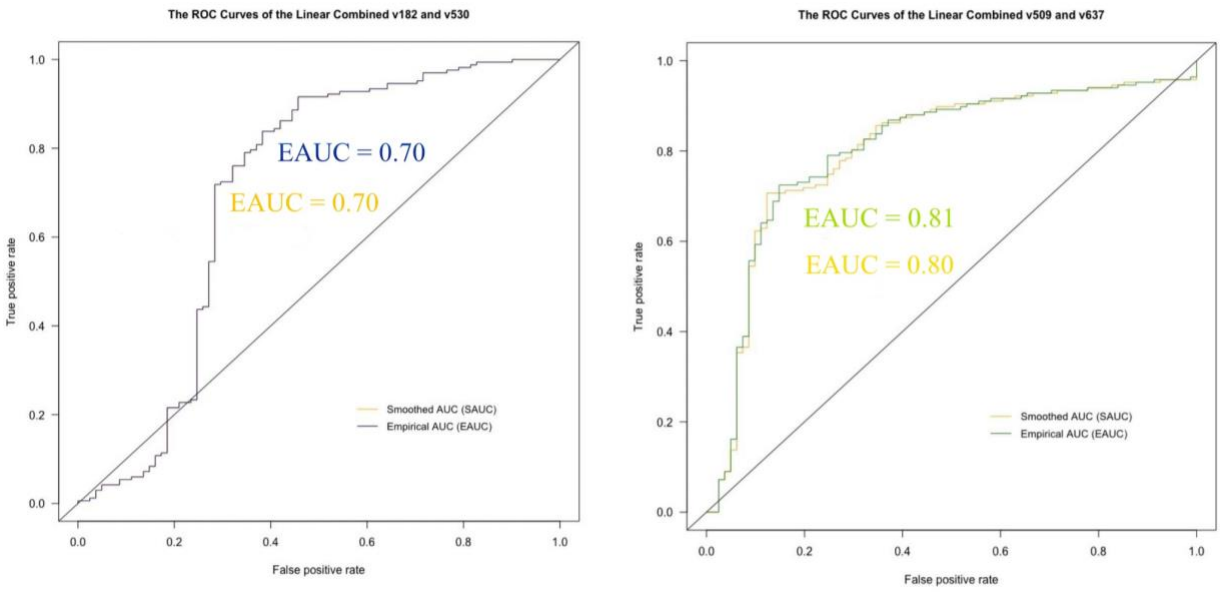

Figure 8 The training ROC curves of the combinations of the selected biomarker pairs maximizing the SAUC versus EAUC (the training EAUCs are situated next to the curves).

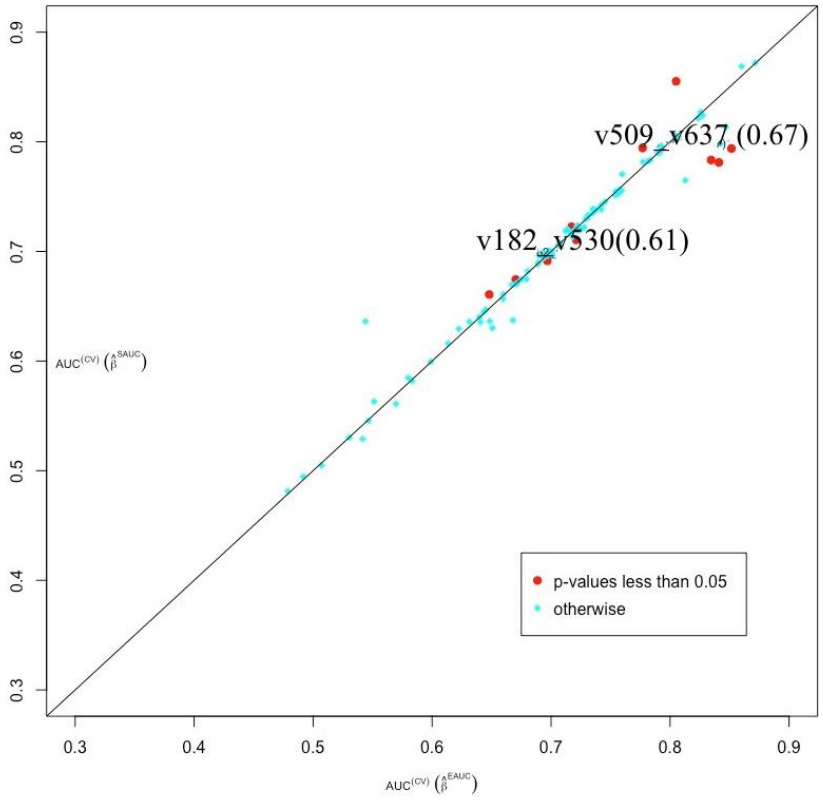

Figure 9 The cross-validated EAUCs of the biomarker pairwise combinations maximizing the SAUC versus EAUC.

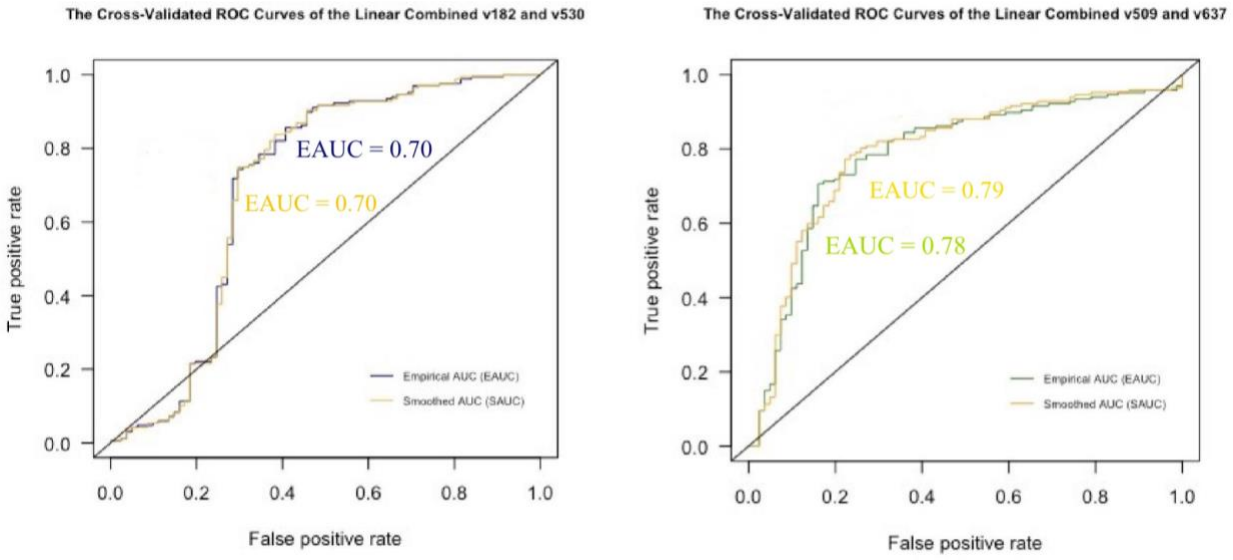

Figure 10 The cross-validated ROC curves of the combinations of the selected biomarker pairs maximizing the SAUC versus EAUC (the cross-validated EAUCs are situated next to the curves).

The cross-validated results shown in Figures 9 demonstrate a fractional advantage of optimization toward EAUC instead of SAUC, especially when AUC > 0.8. We note that for previously considered pairs of markers (v182&v530 and v509&v637), the cross-validated ROC curves remained almost the same (Figure 10).

The results illustrated in Figures 7-10 indicate that the standard smooth approximation of AUC performed generally well for combining two biomarkers. However, there was a potential for suboptimal results when the empirical AUC of the resulting classification score was greater than 0.8. This problem could be exacerbated in the task of combining more than two biomarkers. We considered the results of combining multiple biomarkers in Section 4.1.3.

4.1.1.3 Maximizing $TPF|_{fpf}$ versus AUC

Aside from specific theoretical examples, there is no single linear combination of biomarkers that has a uniformly highest ROC curve (Anderson and Bahadur, 1962). Optimizing a linear combination of markers directly toward $TPF|_{fpf}$ instead of AUC can result in a

classification score with substantially better $TPF|_{fpf}$, at least theoretically (e.g., Figure 1). To evaluate the real-life trends and magnitude of non-trivial improvements in $TPF|_{fpf}$, we considered pairwise linear combinations of 15 biomarkers for prostate cancer (105 pairs overall) optimized using a simple grid search toward the empirical AUC (EAUC) and empirical $TPF|_{fpf=0.1}$ (ETPF).

Figure 11 demonstrates the training estimates of gains in ETPF for the 105 pairs of biomarkers. As is expected in training data, combinations that maximize ETPF always achieves higher ETPF than the same biomarkers' combinations that maximize EAUC. Many of the substantial gains occurred in scenarios where the AUC-maximizing combinations resulted in the ROC curves with a non-concave region ("improperness," Pan and Metz, 1997) in the range of small fpf values. Moreover, substantial gains in TPF happened in cases where the AUC-maximizing combinations achieved ETPF close to 0.5.

The above two types of scenarios are illustrated in Figure 12 for the pairs of biomarkers v354&v530 and v354&v637. Figure 12 shows the ROC curves for the individual biomarkers and their combinations maximizing EAUC and ETPF. The biomarker pair, v354&v530, illustrates the scenario where the EAUC-maximizing linear combination results in an empirical ROC curve residing below the diagonal line in the range of small fpf values. Under such a scenario, optimizing the biomarker combination directly toward ETPF gives rise to a rather substantial gain (increasing ETPF from 0.04 to 0.48). The biomarker pair, v354&v637, illustrates another scenario where the empirical ROC curves of both combinations are almost proper in the range of interest, however, optimization directly toward ETPF leads to a sizable gain by sacrificing performance outside of the range of interest (increasing ETPF from 0.39 and 0.58).

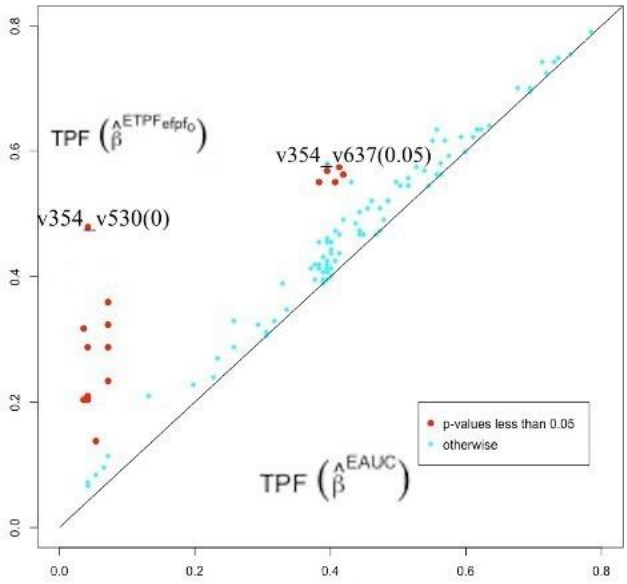

Figure 11 The training ETPFs of the biomarker pairwise combinations maximizing the ETPF versus EAUC.

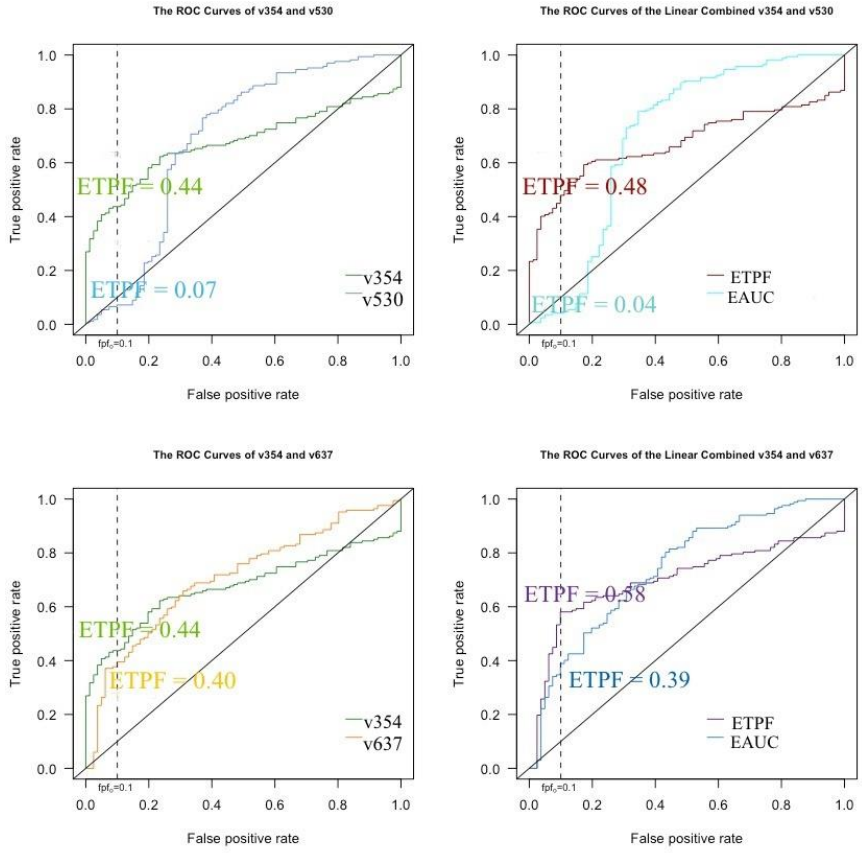

Figure 12 The training ROC curves of the combinations of the selected biomarker pairs maximizing the EAUC versus ETPF.

To alleviate possible biases in the training estimates of TPF-gains shown in Figure 11, we implemented a 10-fold cross-validation procedure (described in Section 3.1.1). Figure 13 shows the cross-validated estimates of empirical $TPF|_{fpf=0.1}$ (ETPF) for linear combinations of biomarkers maximizing EAUC and ETPF (thereby presenting the bias-adjusted version of Figure 11). In contrast to Figure 11, some points in Figure 13 fell below the diagonal, indicating that, for those pairs of biomarkers, optimization directly toward ETPF led to worse cross-validated ETPF than optimization toward EAUC. Substantial gains in ETPF persisted in cases where the EAUC-maximizing combinations resulted in irregular ROC curves. In particular, the previously considered pair of biomarkers, v354&v530, registered approximately the same gain as shown by training estimates (i.e., increasing the cross-validated estimates of ETPF from 0.03 to 0.46, Figure 14). However, the cross-validated gains appeared less substantial than their training estimates in scenarios where the EAUC-maximizing combinations resulted in more regular ROC curves. For example, the previously considered pair of biomarkers, v354&v637, had a noticeably smaller cross-validated gain (i.e., increasing the cross-validated estimates of ETPF from 0.38 to 0.52, Figure 14). Overall, Figure 13 indicates that, when optimizing two biomarkers toward EAUC led to ETPF greater than 0.2, optimization toward TPF did not offer substantially better results. These observations could indicate the instability of maximizing ETPF in small training sets (e.g., during the cross-validation, data are divided into small training sets).

To investigate the possible reasons for suboptimal results of the TPF-based optimization, we considered the relationship between the maximum training ETPF and the difference in the cross-validated estimates of ETPF from optimization toward ETPF and EAUC. Figure 15 demonstrates that optimization toward EAUC achieved substantially better cross-validated ETPF for the pairs of biomarkers that achieved high ETPF in training. The most extreme example was

provided by the biomarker pair v653&v831. As shown in the top half of Figure 16, when combined using the entire dataset, the selected pair of biomarkers achieved a high ROC curve. High ROC curves tend to have steep slopes for small fpf, thereby creating a potential for high variability of the ETPF-related estimates, especially in the smaller training sets, which could lead to combinations that have substantially different performance in the testing sets and/or increase the bias of the pooled cross-validated estimates. The bottom left-hand panel of Figure 16 shows that the average cross-validated ROC curves for the combinations of v653&v831 were approximately the same regardless of the targeted optimization. Thus, the apparent lack of advantages in optimization toward TPF over the AUC-based optimization for high-performance classification scores was an artifact of the pooled cross-validation approach, which appeared to affect the former much more than the latter.

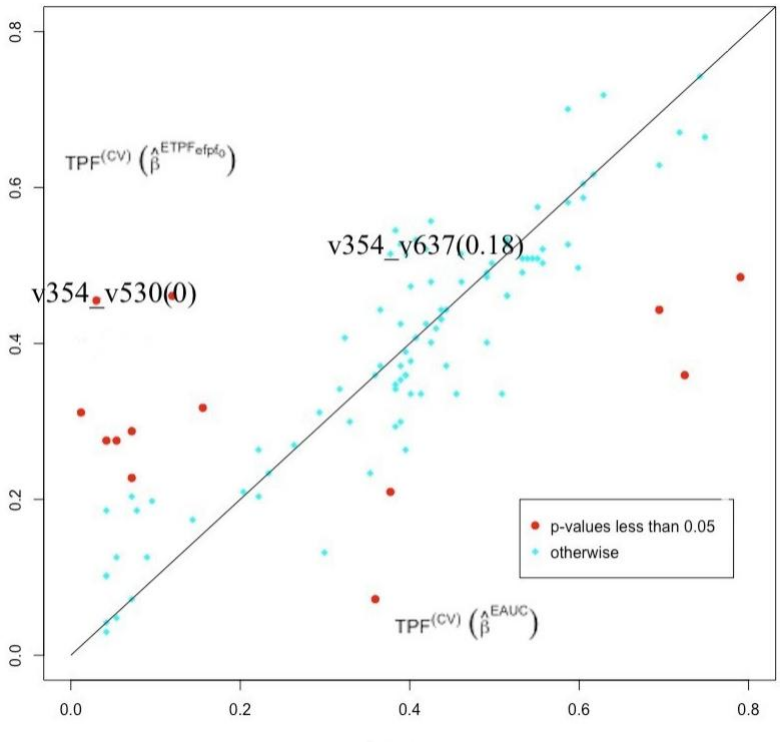

Figure 13 The cross-validated ETPFs of the biomarker pairwise combinations maximizing the ETPF versus EAUC.

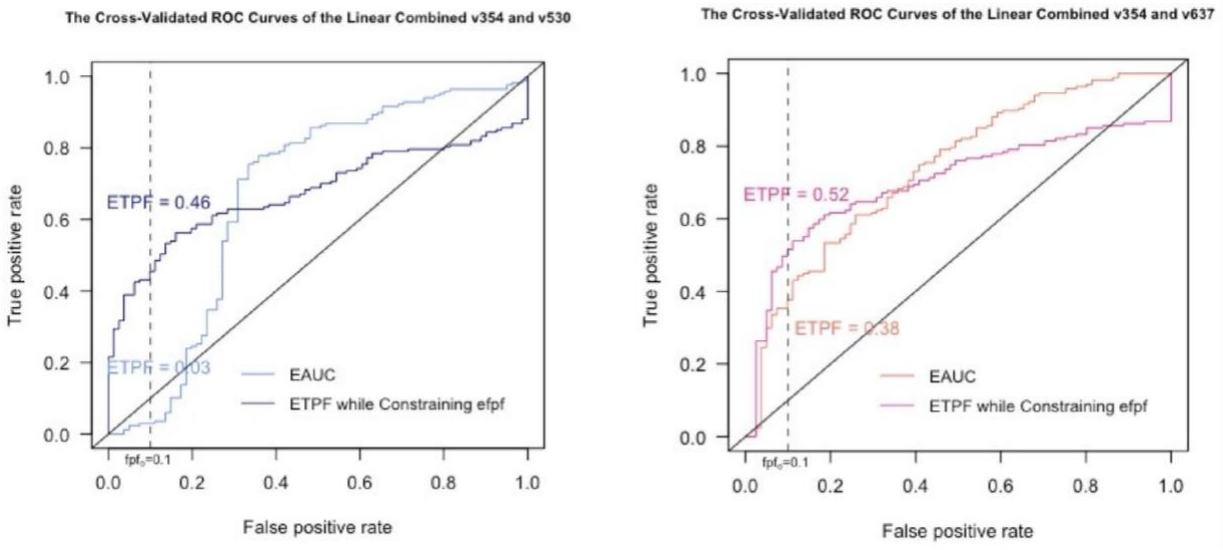

Figure 14 The cross-validated ROC curves of the combinations of the selected biomarker pairs maximizing the ETPF versus EAUC.

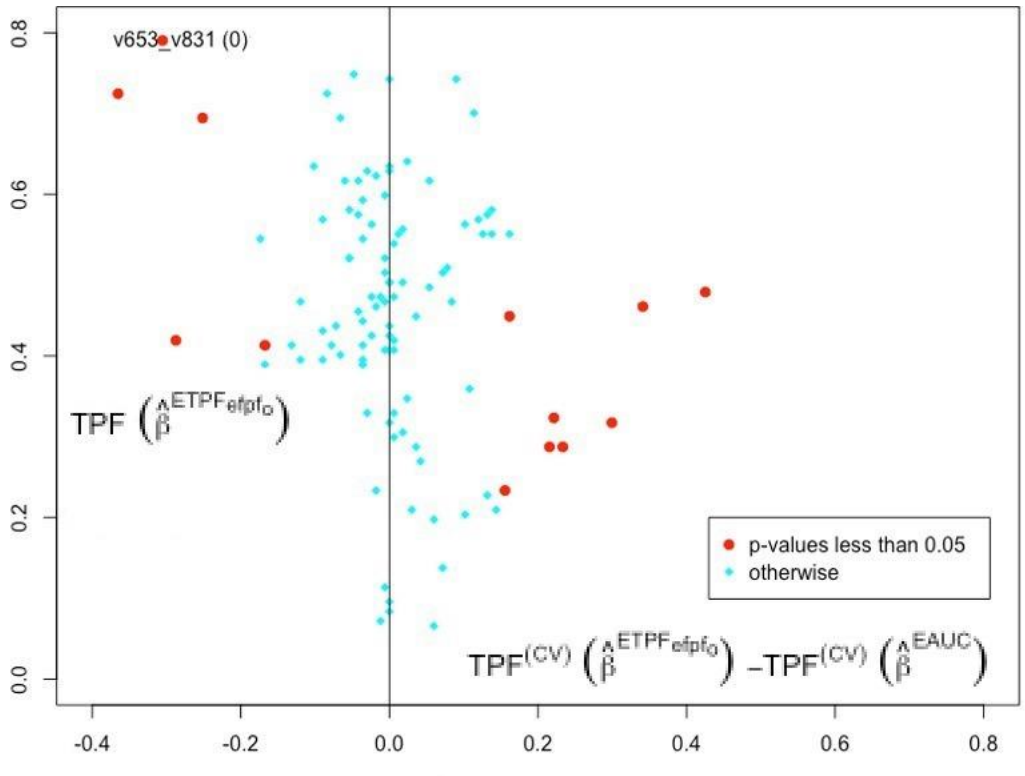

Figure 15 The maximum of training ETPFs versus the difference between the cross-validated ETPF for biomarker combinations maximizing the ETPF versus EAUC.

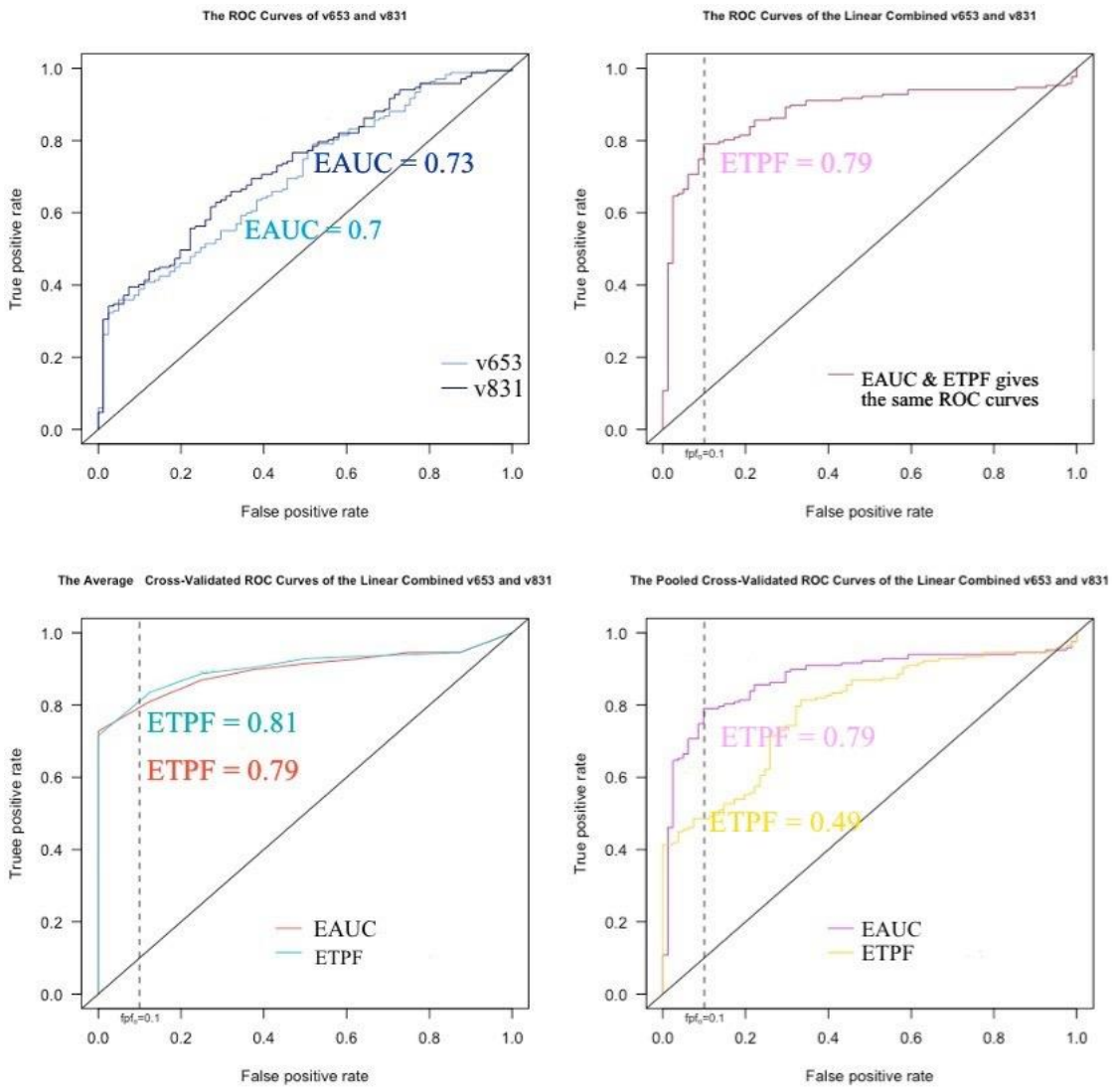

Figure 16 The empirical ROC curves for a selected pair of biomarkers (top left), for combinations maximizing ETPF and EAUC in the entire dataset (top right) as well as average cross-validated (bottom left) and pooled cross-validated (bottom right) ROC curves for the ETPF and EAUC-maximizing combinations.

4.1.1.4 Maximizing a Smooth Approximation to $TPF|_{fpf}$

For later assessments of the gradient-based method, we also examined the performance of biomarker pairwise combinations that maximize a smooth approximation to the $TPF|_{fpf=0.1}$, which we term "STPF" (described in Section 3.1.2). Figure 17 shows that, for many pairs of biomarkers, optimization toward the STPF instead of ETPF resulted in substantially smaller

training ETPF. These advantages of the ETPF optimization could stem from the properties of the considered smooth approximation (e.g., no training ETPF was larger than 0.57 under the STPF-based optimization) or/and could be due to the possible optimism of the training estimates. Comparison of the ROC curves in Figures 18, 20, and 12 (bottom right) for the previously considered pair of markers v354&v637 suggests that maximizing the STPF leads to the ROC curve similar to that resulting from maximizing the EAUC.

The cross-validated estimates summarized in Figure 19 reaffirm substantial advantages of maximizing the ETPF versus STPF. However, this deficiency of the smooth approximation could be overcome by the advantages of simultaneous optimization in the task of combining multiple biomarkers (considered in the next section).

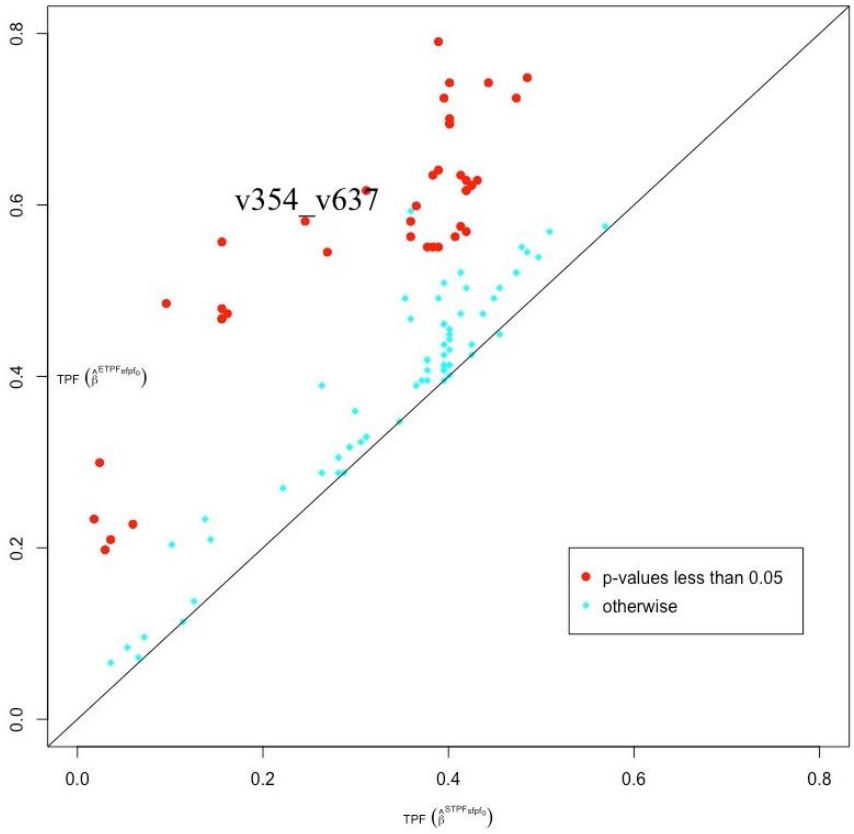

Figure 17 The training ETPFs of the biomarker pairwise combinations maximizing the ETPF versus STPF.

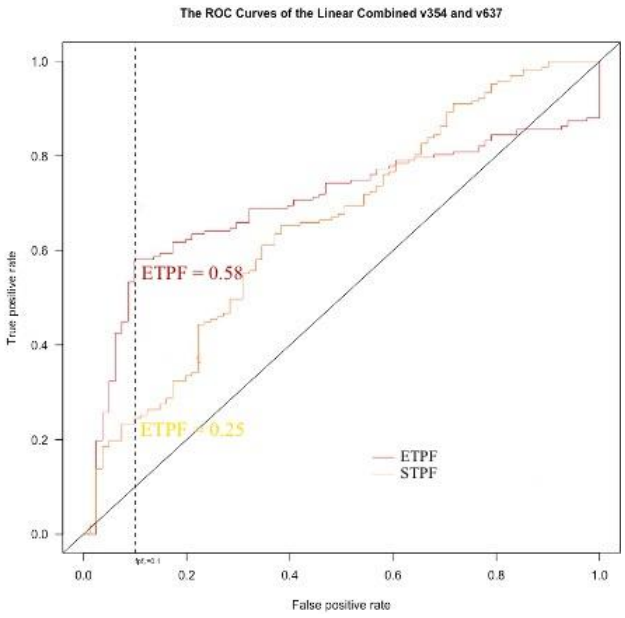

Figure 18 The training ROC Curves of the combinations of the selected biomarker pair maximizing the ETPF versus STPF.

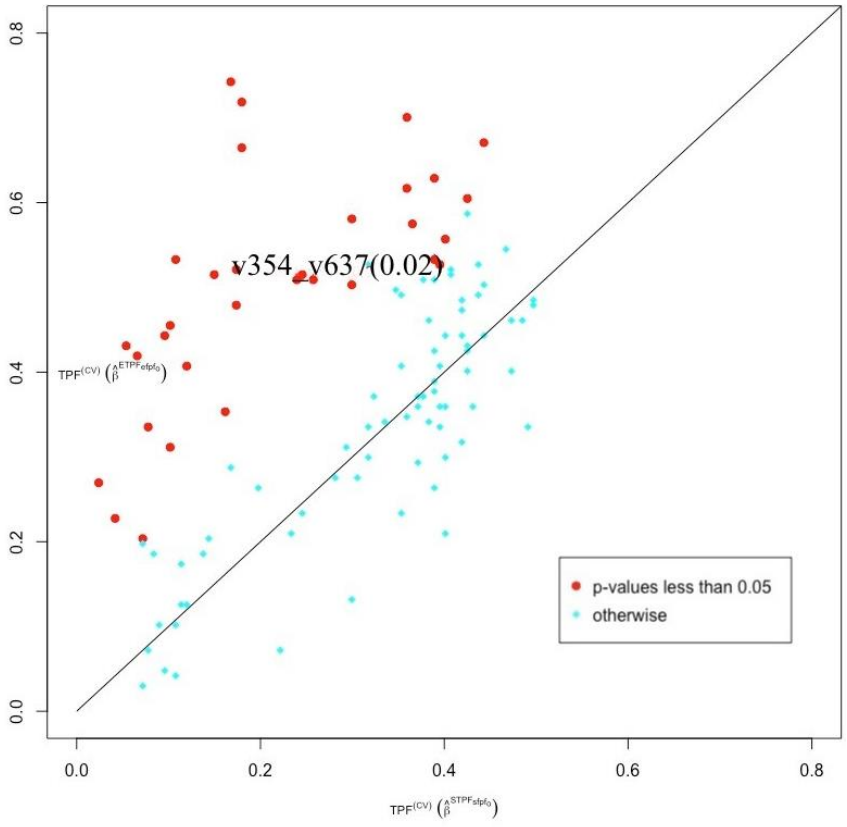

Figure 19 The cross-validated ETPFs of the biomarker pairwise combinations maximizing the ETPF versus STPF.

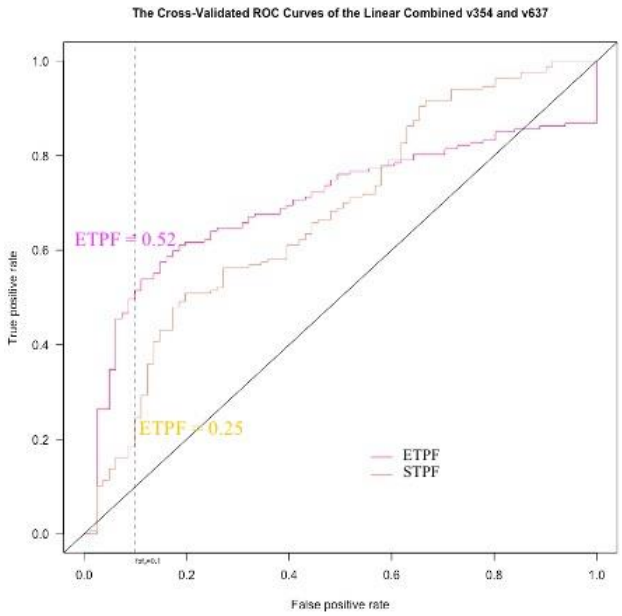

Figure 20 The cross-validated ROC curves of the combinations of the selected biomarker pair maximizing the ETPF versus STPF.

4.1.2 Combinations of Multiple Markers

In this section, we explore linear combinations of markers estimated by methods of optimizing specific ROC characteristics ("ROC-focused methods") and other general objective functions using the dataset of prostate cancer biomarkers. In addition to the methods of optimizing linear combinations of markers, we also assessed the performance of random forests (the R package, randomForest, was used for the assessment), which belongs to a class of non-linear methods for simultaneously combining multiple markers. Due to the flexibility of random forests, the performance of this non-linear method is a useful benchmark that can be deemed as an approximation to the best possible classification accuracy for a given set of biomarkers. Table 2 summarizes the area under the empirical ROC curve (EAUC) and the empirical $TPF|_{fpf=0.1}$ (ETPF) estimated by all the considered approaches for the entire dataset (i.e., re-substitution, or

training estimates) as well as the corresponding cross-validated (CV) estimates obtained by the pooled and average cross-validation approaches (as described in Section 3.1.1).

In terms of the global AUC, all ROC-focused methods that optimize a linear combination of markers attained training EAUC as high as the non-linear combination constructed by random forests (EAUCs of 0.90-0.92 vs. 0.90). The fractionally highest training EAUC was attained by the maxTPR approach (0.92, 95% CI: 0.88-0.95). Overall, (perhaps due to the optimism of the re-substitution estimation), the training EAUC estimates created an impression of similar performance among all the considered approaches. The CV estimates, however, revealed more differences. In particular, the highest pooled-CV estimate of AUC for a linear combination was attained by the sequential AUC-based optimization, which was only fractionally worse than that by random forests (EAUCs of 0.87 vs. 0.91). The second-best pooled-CV estimate of AUC was achieved by ridge regression (EAUC = 0.85) with the other approaches leading to substantially lower values (0.56-0.77). The high value of AUC achieved by the sequential AUC-based optimization was reaffirmed by its average-CV estimate (0.88 vs. 0.91 by random forests). The maxTPR approach resulted in a slightly higher average-CV estimate of AUC (0.90, 95% CI: 0.79-0.90). This result could be caused by the advantages of the simultaneous optimization coupled with a general association between the $TPF|_{fpf}$ and AUC indices. The advantages of the simultaneous optimization were also supported by the performance of the generalized linear models (in particular, both lasso and ridge regression outperformed the stepwise logistic regression). Perhaps as a result of suboptimality observed for pairwise combinations, the performance of the SAUC-based optimization was substantially worse than that of the sequential AUC-based optimization in terms of both the pooled cross-validation (EAUCs of 0.56 vs. 0.87, $p < 0.001$) and the average cross-validation (EAUCs of 0.64 vs 0.88, $p < 0.001$) procedures.

In terms of a more specific performance characteristic represented by $TPF|_{fpf=0.1}$, all ROC-focused methods attained high values of training ETPFs (0.81-0.88), which were substantially larger than those obtained by the standard statistical models (0.58-0.77). The maxTPR approach attained the highest training ETPF (0.88), with the sequential TPF-based and AUC-based optimizations trailing closely behind (ETPFs of 0.86 and 0.83, respectively). However, the pooled-CV estimates of $TPF|_{fpf=0.1}$ were substantially smaller for maxTPR and the sequential TPF-based optimization than for the sequential AUC-based optimization and for ridge regression (0.45 and 0.37 vs. 0.73 and 0.68). This anomaly appeared to be in part caused by the bias of the pooled-CV estimates of ROC-related indices (Airola et al., 2011). Indeed, maxTPR had the highest average-CV estimate of $TPF|_{fpf=0.1}$ (0.76) among all the considered approaches, with the sequential AUC-based optimization leading to the second-largest estimate (0.65, $p < 0.001$). The rest of the methods, including the sequential-based optimization, had substantially lower averaged-CV estimates of $TPF|_{fpf=0.1}$ (ranging from 0.27 to 0.57).

Table 2 Training and testing performance characteristics for linear combinations of markers optimized using different approaches.

Methods of optimizing combinations of markers	Performance Measures					
	Training		Cross-validated			
			Pooled		Average	
	EAUCs	ETPFs	EAUCs	ETPFs	$\overline{EAUCs}$	$\overline{ETPFs}$
Maximal AUC (Sequential)	0.90	0.83	0.87	0.73	0.88	0.65
Maximal $TPF|_{fpf}$ (Sequential)	0.91	0.86	0.75	0.45	0.77	0.55
Logistic Regression (Seqential)	0.90	0.77	0.63	0.27	0.67	0.47
Lasso (Simultaneous)	0.85	0.58	0.77	0.38	0.85	0.54
Ridge Regression (Simultaneous)	0.88	0.69	0.85	0.68	0.86	0.57
Maximal SAUC (Simultaneous)	0.91	0.81	0.56	0.08	0.64	0.27
Maximal STPF (Simultanous)	0.92	0.88	0.65	0.37	0.90	0.76
Random Forests* ***(Simultaneous)***	0.90	0.76	0.91	0.81	0.91	0.65

** Random forests estimate a non-linear combination that provides an approximation to the globally optimal combination of biomarkers.*

Table 3 summarizes standardized coefficients of biomarkers in linear combinations optimized by different approaches. The combination coefficients were standardized by the standard deviation of the individual biomarker values to facilitate the comparison within each approach and rescaled to the norm of 1 to facilitate the comparison between the approaches. Hence, coefficients can be interpreted as weights representing the contribution of individual biomarkers to the resulting classification score. All ROC-focused combinations included large weights for biomarker v653, and most methods allowed for the substantial contribution of biomarker v831, which had the best empirical AUC among all individual biomarkers (Table 1). Standard statistical methods (that optimize more general objective functions) heavily weighted biomarker v831, yet largely ignored the marker v653, indicating its specific relevance to the ROC measures. However, unlike other ROC-focused approaches, SAUC incorporated biomarker v652 instead of the univariately most discriminative v831. Similar variations seemed to have contributed to the suboptimal performance of the SAUC-based optimization during the assessments of cross-validation.

Overall, the presented results corroborate the expectations that training estimates of performance levels can be misleading, with little difference noticeable among the optimization methods, and that a standard pooled cross-validation approach can substantially misrepresent the performance of optimized classifiers (at least in the datasets of similar size). The latter is particularly evident for optimization driven by more focused characteristics, such as $TPF|_{fpf}$, which might be more sensitive to changes in the training sets (especially with a small sample size). The results also indicate that sequential procedures might work well for maximizing global measures, such as AUC, but are less stable for optimizing more specific objective functions, such as $TPF|_{fpf}$. Moreover, the use of smooth approximation had a different impact on the considered

optimization methods. It was quite beneficial for simultaneous optimization toward a specific objective function represented by $TPF|_{fpf}$, but apparently detrimental for optimization toward more global ROC indices represented by AUC (with SAUC leading to an undesirable EAUC). The latter part of the observation was reaffirmed by the results given by optAUC (0.65 and 0.37 for EAUC and ETPF, respectively), which entails the implementation of an approximated leave-one-out cross-validation and a smooth approximation by the sigmoid function for the estimation of AUC (see Huang et al., 2011, for detail). The standardized coefficients of biomarkers in linear combinations optimized by optAUC, however, indicate a pattern of biomarker contributions different from those of other considered ROC-focused methods. Lastly, at least in the current dataset, the simultaneous optimization toward the STPF by maxTPF gave rise to very competitive results in terms of both ETPF and EAUC.

Table 3 Standardized coefficients of biomarkers in linear combinations optimized by different approaches.

			Methods of optimizing linear combination of markers				
Biomarkers	**Maximal AUC (Sequential)**	**Maximal $TPF|_{fpf}$ (Sequential)**	**Lasso**	**Ridge Regression**	**Maximal SAUC**	**optAUC**	**Maximal STPF (maxTPR)**
v30	0.03	0.03	0.19	0.26	0.02	-0.09	0.1
v93			-0.09	-0.15	-0.01	0	-0.03
v182				0.02	0	-0.33	0.07
v354			-0.34	-0.4	-0.05	0.6	-0.11
v365				0.17	0.01	-0.09	0.1
v426			-0.21	-0.22	-0.03	0.19	0.08
v427	-0.02	-0.04	-0.11	-0.18	0	0.16	-0.3
v509	-0.02		0	-0.35	-0.03	0.31	-0.28
v530			0.15	0.24	0.02	-0.14	0.09
v637				-0.1	-0.01	0.24	0.05
v652	0.02	0.02		0.17	0.69	0.22	-0.12
v653	-0.66	-0.66		0.08	-0.71	0.22	-0.5
v741				0.34	-0.05	0.26	0.32
v831	0.75	0.75	0.87	0.4	0.07	0.25	0.43
v877	0.02		0.12	0.37	0.1	0.25	0.48

5.0 Summary and Discussion

This work explored the real-life gains from task-oriented optimization of biomarker combinations, with the primary focus on explicit and interpretable linear combinations. In our exploration, we used the dataset of 15 quantitative biomarkers from the protein mass spectrometry study (Yasui, et al., 2003), which had been used for illustration in a fundamental work advocating classification-oriented optimization toward AUC (Pepe et al., 2006). In light of the need for high sensitivity that is constrained by the necessity to limit false positive results for practical application (e.g., screening a large population), we considered optimization toward the highest empirical estimate of $TPF|_{fpf=0.1}$ (ETPF) as the ultimate task-oriented optimization (termed optimization toward TPF for brevity). We compared optimization toward TPF versus AUC for sequential and simultaneous combinations of 15 biomarkers (altogether and in pairs). The gain from optimization toward TPF was assessed by the magnitude of the increase in the ETPF obtained from the optimal combination of biomarkers. We also considered the performance of more general statistical approaches as benchmarks. This part of the investigation included reiteration and expansion of the previous illustrations by Pepe et al., 2006.

The results indicate that task-oriented optimization can have substantial advantages over the global AUC optimization for sequential (pairwise included) and simultaneous combinations of biomarkers. Substantial gains in the ETPF occur when individual biomarkers have very improper (i.e., non-concave) and differently shaped ROC curves as well as roughly proper ones. Importantly, the non-linear combinations of biomarkers optimized by the conventional method of random forests did not offer substantial improvement in the ETPF, indicating that optimizing more

complicated combinations of biomarkers may still benefit from the use of the task-specific function for internal optimization.

To better characterize the possible advantages of using task-specific objective functions, several optimization algorithms were considered. The most robust grid search algorithm was used for optimizing biomarker pairwise combinations as well as sequential combinations of multiple biomarkers that maximize the empirical AUC (EAUC) and $TPF|_{fpf=0.1}$ (ETPF). As for simultaneous optimization of multiple biomarkers, we used the methods of smooth approximations to AUC and $TPF|_{fpf=0.1}$ (Fong, et al., 2016 and Meisner et al., 2017). To determine the effects of the implemented smooth approximations, the use of the smoothed AUC (SAUC) and the smoothed TPF (STPF by maxTPR) as objective functions was evaluated in the context of biomarker pairwise combinations optimized through gradient-based algorithms. The results indicate that the implication of using smooth approximations hinges on the objective function as well as the number of combined markers. Although there appeared no advantages of using the maxTPR approach to optimizing pairs of biomarkers, it led to competitive results from simultaneous optimization of all 15 biomarkers. Opposite results were observed for SAUC from simultaneous optimization of all 15 biomarkers. Interestingly, SAUC did relatively well at optimizing pairs of biomarkers in contrast to the performance of maxTPR in a similar situation. At least part of what we have observed so far is related to the deficiencies in the numeric optimization of the resulting objective functions, which might be alleviated by using more advanced optimization algorithms (e.g., a difference of convex functions algorithm by Fong et al., 2016). Overall, the maxTPR approach for simultaneous optimization is critical to tackling the task of combining multiple markers for classification, as the performance of sequential optimization toward TPF was not desirable.

In analyzing the gains using the task-specific objective functions, we used a 10-fold cross-validation. Since the conventional "pooled" cross-validation could lead to downward-biased results (e.g., Airola et al., 2011) for optimization toward ROC-related indices, we also considered the less biased (albeit more variable) "average" cross-validation, which uses the average of fold-specific ROC curves instead of the ROC curve for the pooled data. The results indicate that the negative bias of the pooled cross-validation could be substantial, especially for the task of combining multiple markers. Moreover, the magnitude of the bias depends on the objective function, with the TPF-based optimization being much more affected than the global AUC-based optimization. It is worth noting that the magnitude of the bias from the pooled cross-validation was large enough to make the performance of the maxTPR approach for simultaneous optimization appear much worse than that of the sequential optimization toward AUC.

In conclusion, the results of our investigation indicate that substantial gains can be achieved by optimizing a combination of biomarkers toward a task-specific objective function based on ROC-related indices (e.g., $TPF|_{fpf}$ for tasks requiring limited false positive, such as cancer screening of a large asymptotic population). Sustaining the benefits of optimization toward task-specific objective functions for combining multiple markers requires solving them simultaneously, of which the solution is conveniently offered by the R package "maxTPR" (Meisner et al., 2017). Further development on the topic is warranted by incorporating task-specific objective functions in non-linear approaches such as random forests and neural networks.

Appendix A Partial Area Under the Curve

Appendix A.1 Combinations of Two Biomarkers: Maximizing $pAUC(0, fpf)$ versus $TPF|_{fpf}$

The use of $TPF|_{fpf}$ as a target for optimizing biomarker combinations focuses on the sensitivity of decisions with low fpf (i.e., high specificity), which is relevant for many practical tasks. A related but more global, index is provided by $pAUC(0, fpf)$,which is the area under the range of low fpf. Like $TPF|_{fpf}$, $pAUC(0, fpf)$ has also been proposed as a target for optimizing biomarker combinations (e.g., Komori and Eguchi, 2010 and Wang and Chang, 2011). By summarizing a larger part of the ROC curve, the pAUC-based optimization might be more stable, yet it is not as specific as the TPF-based optimization. It is unclear whether any meaningful gains could be obtained by using $TPF|_{fpf}$ as an optimization target instead of $pAUC(0, fpf)$. To explore the possible real-life gains, we optimized pairwise linear combinations of 15 prostate cancer biomarkers toward the empirical $pAUC(0, 0.1)$ (EpAUC) and $TPF|_{fpf=0.1}$ (ETPF).

Figure 21 illustrates the training estimates of gains in the ETPF for all the biomarker pairs. As is expected in training data, biomarker combinations that maximize the ETPF always achieves higher ETPF than the same biomarkers' combinations that maximize the EpAUC. However, almost none of the gains was substantial with only one exception of the biomarker pair, v509&v637. Figure 22 indicates that for v509&v637, optimization toward ETPF achieved high ETPF by sacrificing performance regarding EpAUC. Figure 23 (the cross-validated version of Figure 21) shows substantial gains in the cross-validated ETPF for many pairs of biomarkers and

virtually no instances of pAUC-based optimization resulting in substantially higher ETPF. Comparing with Figure 13, the observed results indicate that pAUC-based optimization did not offer the improvements in stability as AUC-based optimization did.

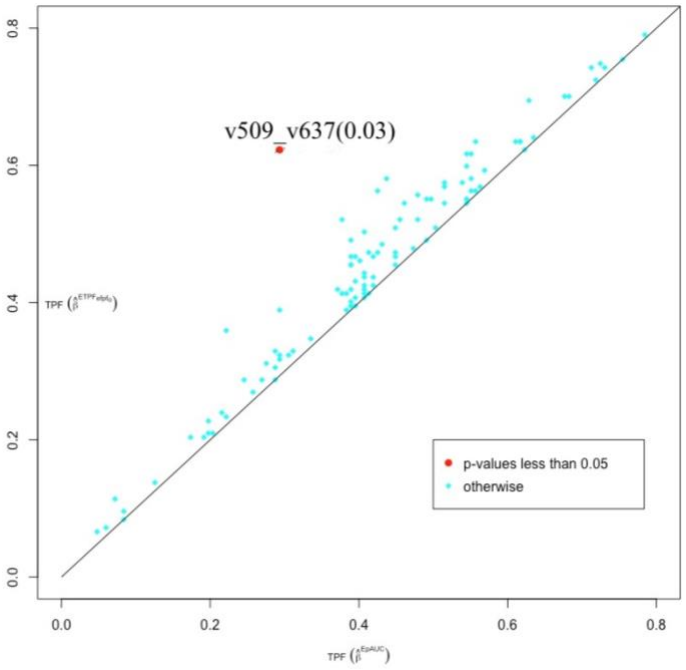

Figure 21 The training ETPFs of the biomarker pairwise combinations maximizing the ETPF versus the EpAUC.

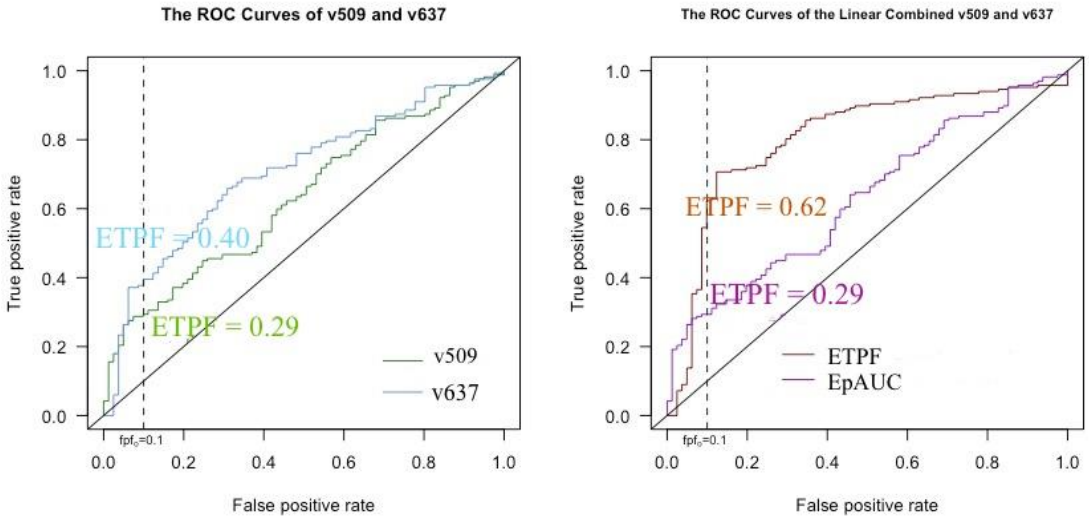

Figure 22 The training ROC curves of the combination of the selected biomarker pair maximizing the ETPF versus EpAUC.

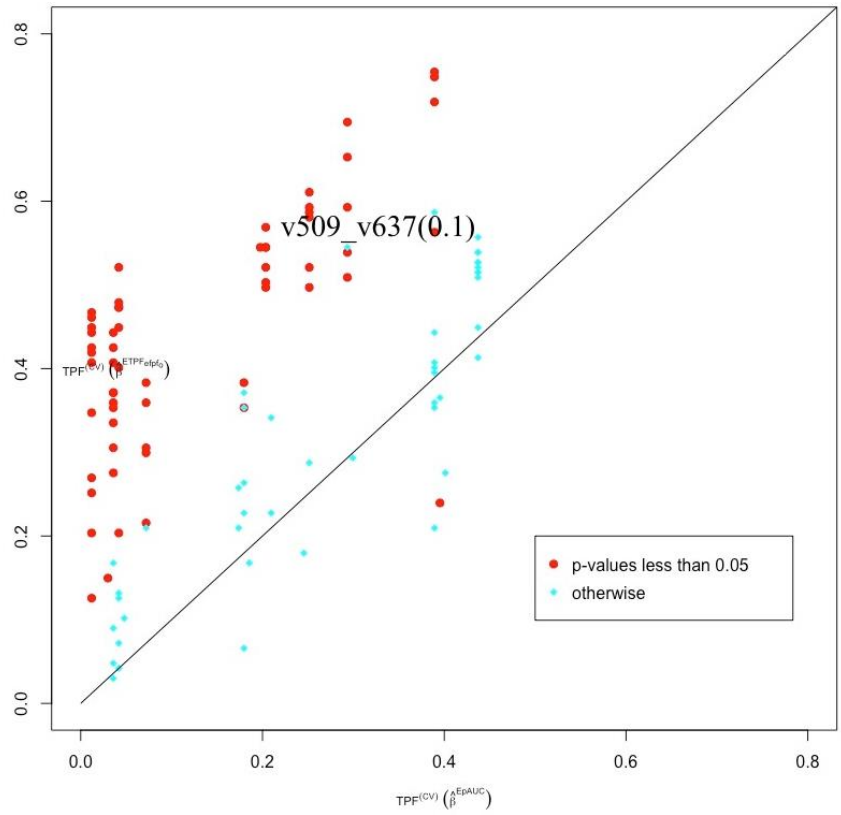

Figure 23 The cross-validated ETPFs of the biomarker pairwise combinations maximizing the ETPF versus EpAUC.

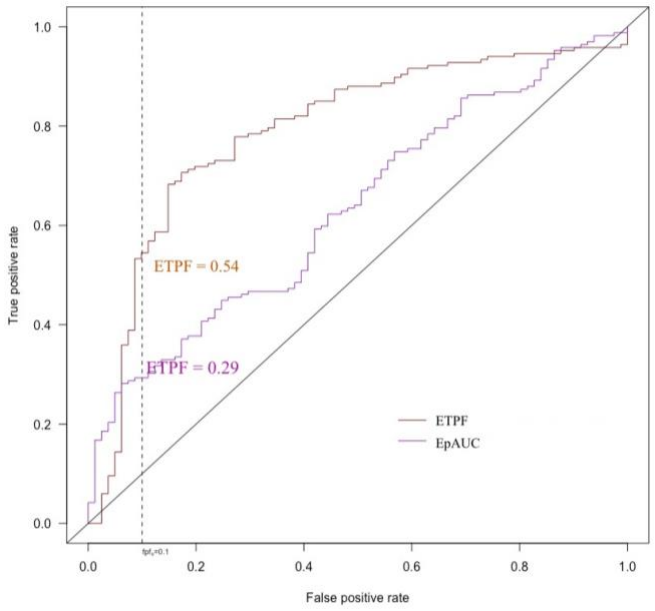

Figure 24 The cross-validated ROC curves of the combinations of the selected biomarker pair maximizing the ETPF and EpAUC.

Appendix A.2 Combination of Two Biomarkers: Maximizing a Smooth Approximation to $\boldsymbol{pAUC(0, fpf)}$

Like other ROC-related objective functions, a smooth approximation to $pAUC(0, fpf)$ (SpAUC) was proposed as a more regular objective function than the empirical one. We estimated the empirical $TPF|_{fpf=0.1}$ (ETPF) achieved by biomarker pairwise combinations that maximize a smooth approximation to $pAUC(0, fpf)$ (SpAUC). Figure 25 shows that optimization toward SpAUC resulted in slightly larger training ETPFs than the EpAUC-based optimization did, indicating a possible benefit of using a smooth approximation. As for the previously consider pair of biomarkers v509&v637, the SpAUC-based and ETPF-based optimizations led to similar training ROC curves (Figures 26 and 22). Furthermore, optimization toward the SpAUC versus EpAUC appeared to have similar cross-validated gains in the ETPF as optimization toward the ETPF versus EpAUC did (Figures 27 and 23).

Overall, pAUC is an optimization target which lies in between AUC and $TPF|_{fpf}$ and inherits strengths and weaknesses of both albeit to a lesser degree. This property might be the reason that smooth approximation worked relatively well with $pAUC(0, fpf)$ when used to optimize a pair of biomarkers, but not with AUC and $TPF|_{fpf}$ under similar settings.

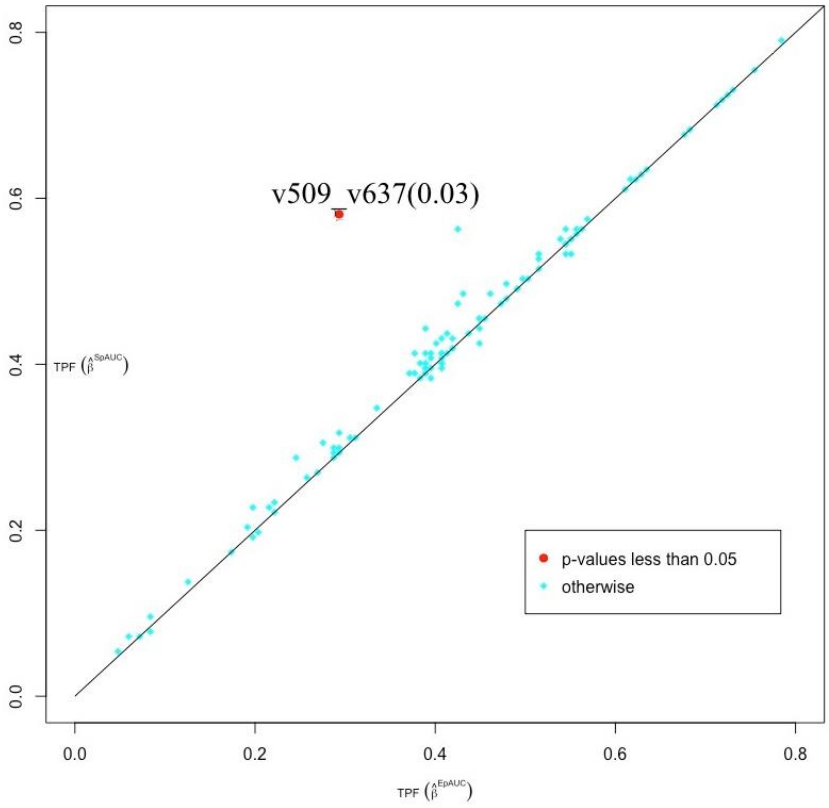

Figure 25 The training ETPFs of the biomarker pairwise combinations maximizing the SpAUC versus EpAUC.

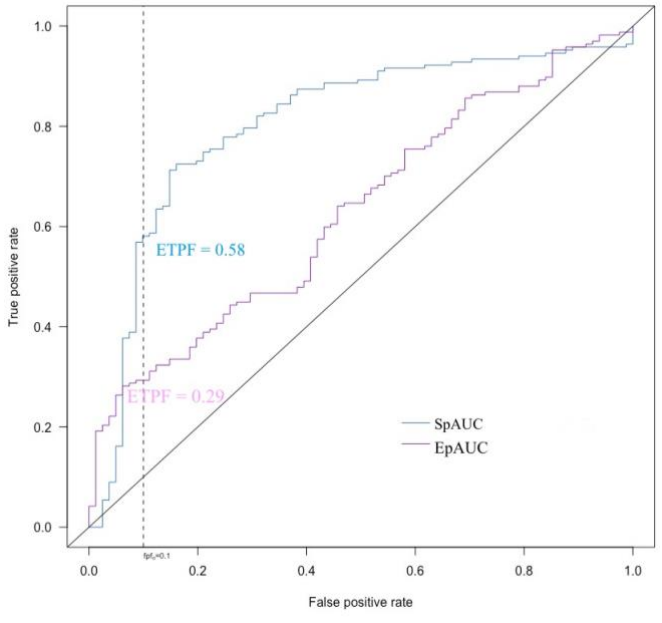

Figure 26 The training ROC Curves of the combinations of the selected biomarker pair maximizing the SpAUC and EpAUC.

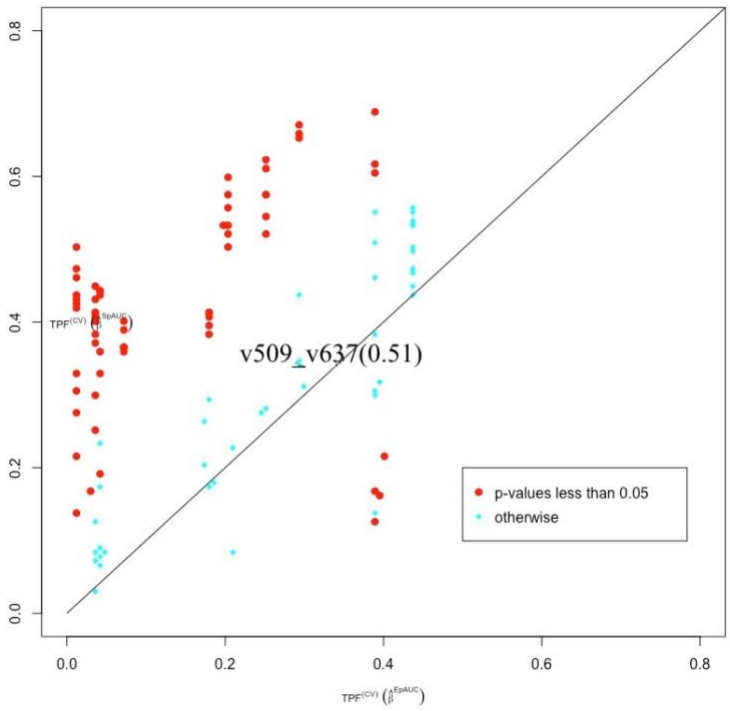

Figure 27 The cross-validated ETPFs of the biomarker pairwise combinations maximizing the SpAUC versus EpAUC.

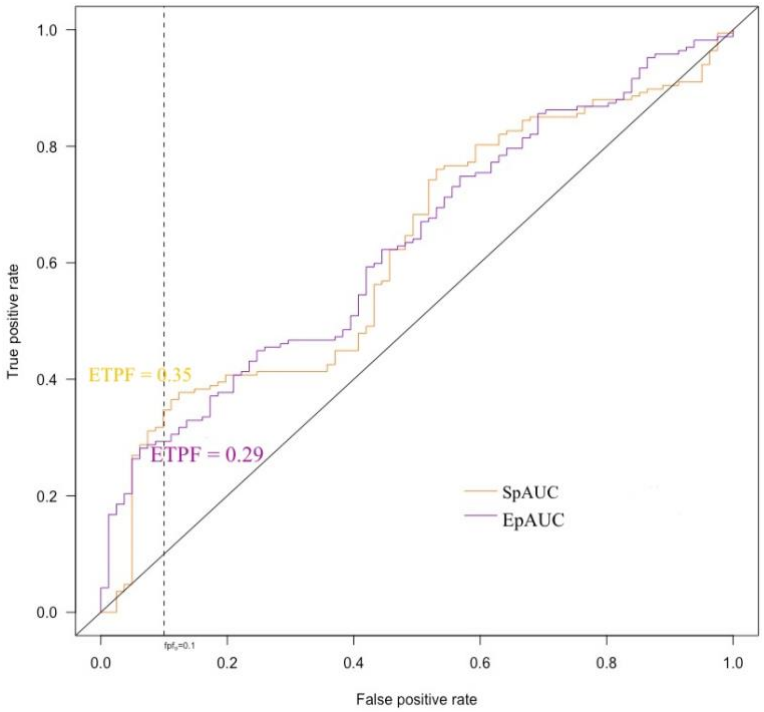

Figure 28 The cross-validated ROC curves of the combinations of the selected biomarker pair maximizing the SpAUC and EpAUC.

Appendix B R Code

Appendix B.1 Data Import and Transformation

```
## set up the working directory
setwd("~/Desktop/Thesis")

## load all the R packages used in the analysis
packages_required <- c("caret","glmnet","maxTPR", "aucm",
                       "pROC","clinfun","tidyverse","AUCRF","randomForest")
lapply(packages_required, require, character.only = TRUE)

## import the prostate cancer dataset
prostate.cancer <- read.csv("yy_biom62dat.csv")

## transform values of some of the biomarkers for the purpose of our analysis
biomarkers <- prostate.cancer[,-1]
biomarkers_t <- biomarkers
biomarkers_t$v30 <- -biomarkers$v30
biomarkers_t$v182 <- -biomarkers$v182
biomarkers_t$v354 <- -biomarkers$v354
biomarkers_t$v365 <- -biomarkers$v365

## create two data frames that contain cases only and controls only
case_1 <- biomarkers_t[biomarkers_t$case==1, -1]
cont_0 <- biomarkers_t[biomarkers_t$case==0, -1]
```

Appendix B.2 Optimization of Two Biomarkers

```
## function to calculate the empirical area under the ROC curve
eAUC <- function(v0, v1) {
  n0 <- length(v0)
  n1 <- length(v1)
  sum_ef=0
  for (i in 1:n0) {
    for (j in 1:n1) {
      if (v0[i] < v1[j]) {
        sum_ef = sum_ef+1
```

```
      } else {
        sum_ef = sum_ef
      }
    }
  }
  auc_ef = round(sum_ef/(n0*n1), 4)
  return(auc_ef)
}

## optimization of two biomarkers toward logistic likelihood
AUCe_lr <- vector("numeric")
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      dat <- data.frame(case=biomarkers_t$case, marker_1=biomarkers_t[,s+1],
marker_2=biomarkers_t[,t+2])
      fit <-  glm(case ~ marker_1 + marker_2, dat, family="binomial")
      b1 <- fit$coef[2]
      b2 <- fit$coef[3]
      v1 <- b1*m11 + b2*m21
      v0 <- b1*m10 + b2*m20
      AUCe_lr <- c(AUCe_lr, eAUC(v0, v1))
  }
  }
}
AUCe_lr

## optimization toward the empirical area under the ROC curve

max.gamma <- vector("numeric")
gamma_angle <- seq(1, 360, by=0.5)
eauc2 <- vector("numeric")
AUCe_do <- vector("numeric")
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      for (g in 1:length(gamma_angle)) {
        b1 <- cos((gamma_angle[g]*pi)/180)
        b2 <- sin((gamma_angle[g]*pi)/180)
        v1 <- b1*m11 + b2*m21
        v0 <- b1*m10 + b2*m20
        eauc2 <- c(eauc2, eAUC(v0, v1))
```

```
      }
      AUCe_do <- c(AUCe_do, eauc2[which.max(eauc2)]); max.gamma <- c(max.gamm
a, gamma_angle[which.max(eauc2)])
    }
    eauc2 <- vector("numeric")
  }
}
AUCe_do

## function to calculate the empirical true positive fraction at a given level of specificity
eTPF <- function(fpf0=0.1, v0, v1) {
  n0 <- length(v0)
  n1 <- length(v1)
  v0o = v0[order(v0)]
  v1o = v1[order(v1)]
  sum_ef=0
  for (j in 1:n1) {
    if (v0o[n0-floor(fpf0*n0)] < v1o[j]) {
      sum_ef = sum_ef+1
    } else {
      sum_ef = sum_ef
    }
  }
  tpf_ef = round(sum_ef/n1,4)
  return(tpf_ef)
}

## optimization toward the empirical true positive fraction at a given level of specificitymaxgamm
a_etpf <- vector("numeric")
etpf2 <- vector("numeric")
TPFe_do <- vector("numeric")
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      for (g in 1:length(gamma_angle)) {
        b1 <- cos((gamma_angle[g]*pi)/180)
        b2 <- sin((gamma_angle[g]*pi)/180)
        v1 <- b1*m11 + b2*m21
        v0 <- b1*m10 + b2*m20
        etpf2 <- c(etpf2, eTPF(0.1, v0, v1))
      }
      TPFe_do <- c(TPFe_do, etpf2[which.max(etpf2)]); maxgamma_etpf <- c(maxg
amma_etpf, gamma_angle[which.max(etpf2)])
    }
    etpf2 <- vector("numeric")
  }
```

```
}
TPFe_do

##  function to calculate the empirical partial area under the ROC curve
epAUC <- function(fpf0=0.1, v0, v1) {
  n0 <- length(v0)
  n1 <- length(v1)
  v0o = v0[order(v0)]
  v1o = v1[order(v1)]
  sum_ef1=0
  for (i in (n0-floor(fpf0*n0)+1):n0) {
    for (j in 1:n1) {
      if (v0o[i] < v1o[j]) {
        sum_ef1 = sum_ef1+1
      } else {
        sum_ef1 = sum_ef1
      }
    }
  }
  pauc_ef1 = sum_ef1/(n0*n1)
  sum_ef2=0
  for (j in 1:n1) {
    if (v0o[n0-floor(fpf0*n0)] < v1o[j]) {
      sum_ef2 = sum_ef2+1
    } else {
      sum_ef2 = sum_ef2
    }
  }
  pauc_ef2 = (sum_ef2/n1)*(fpf0-(floor(fpf0*n0)/n0))
  pauc_ef = round(pauc_ef1+pauc_ef2, 4)
  return(pauc_ef)
}

## optimization toward the empirical partial area under the curve
epauc2 <- vector("numeric")
TPFe_pauc <- vector("numeric")
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      for (g in 1:length(gamma_angle)) {
        b1 <- cos((gamma_angle[g]*pi)/180)
        b2 <- sin((gamma_angle[g]*pi)/180)
        v1 <- b1*m11 + b2*m21
        v0 <- b1*m10 + b2*m20
        epauc2 <- c(epauc2, epAUC(fpf0=0.1, v0, v1))
      }
```

```
      b1_op <- cos((gamma_angle[which.max(epauc2)]*pi)/180)
      b2_op <- sin((gamma_angle[which.max(epauc2)]*pi)/180)
      v1_op <- b1_op*m11 + b2_op*m21
      v0_op <- b1_op*m10 + b2_op*m20
      TPFe_pauc <- c(TPFe_pauc, eTPF(0.1, v0_op, v1_op))
    }
    epauc2 <- vector("numeric")
  }
}
TPFe_pauc

## function to generate pair difference
calXdiff<-function(X,Y,d) {
  x_nd<-X[Y==0,]
  x_d<-X[Y==1,]

  n1=sum(Y==1);
  n0=sum(Y==0)
  x_ndr= apply(x_nd, 2, rep, n1)
  x_dr<-apply(x_d,2,rep,each=n0)

  x_diff=x_dr-x_ndr


  x_diff_mul=matrix(0,dim(x_diff),d*d)
  for (p in 1:d) {
    for (q in 1:d) {
      x_diff_mul[,(p-1)*d+q]=as.matrix(x_diff[,p]*x_diff[,q])
    }
  }
  out=list(x_diff,x_diff_mul)
  return(out)
}

## function to standardize the components of a vector according to its norm
normsq<-function(x) {
  return(x/sqrt(sum(x^2)))
}

## generate parameter values for grid search
b1v <- vector("numeric")
b2v <- vector("numeric")
for (g in 1:length(gamma_angle)) {
  b1v <- c(b1v, cos((gamma_angle[g]*pi)/180))
  b2v <- c(b2v, sin((gamma_angle[g]*pi)/180))
}
beta_v <- cbind(b1v, b2v)

## Generate values of the tuning parameter for the smooth approximation
h_sa <- vector("numeric")
```

```
for (s in 1:14) {
  for  (t in 1:14) {
    if (s <= t) {
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      tmp <- biomarkers_t[,c(1, s+1, t+2)]
      dat4anal <- model.frame(tmp[,1]~tmp[,2]+tmp[,3], dat=tmp)
      Y <- dat4anal[,1]
      n1 <- sum(Y==1)
      n2 <- sum(Y==0)
      n <- n1+n2
      markers <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3],dat=tmp)[,-1]
      num_markers <- ncol(markers)
      X1 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==1,])
[,-1,drop=FALSE]
      X2 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==0,])
[,-1,drop=FALSE]
      fit.rlogit  <- rlogit(tmp[,1]~tmp[,2]+tmp[,3], tmp)
      if (fit.rlogit$convergence) {
        beta.init <- fit.rlogit$coef[-1]
      } else {
        beta.init <- rep(1, ncol(tmp)-1)
      }
      Xint <-calXdiff(markers, Y, d=length(beta.init))
      x_diff <- Xint[[1]]
      h_sa <- c(h_sa, n^(-1/2)*sd(drop(x_diff%*%normsq(beta.init))))
    }
  }
}

## optimization toward a smooth approximation to AUC
AUCe_sAUC <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for  (t in 1:14) {
    if (s <= t) {
      counter <- counter + 1
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      tmp <- biomarkers_t[,c(1, s+1, t+2)]
      out <- grid.auc(tmp[,1]~tmp[,2]+tmp[,3], tmp, beta_v, approx.type="norm
al", approx.param=h_sa[counter], lambda=0)
      b1_sAUC <- out$coefficient[1]
      b2_sAUC <- out$coefficient[2]
      v1 <- b1_sAUC*m11 + b2_sAUC*m21
      v0 <- b1_sAUC*m10 + b2_sAUC*m20
```

```
      AUCe_sAUC <- c(AUCe_sAUC, eAUC(v0, v1))
    }
  }
}
AUCe_sAUC

## pairwise optimization toward a smooth approximation to TPF|fpf
TPFe_sTPF <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for  (t in 1:14) {
    if (s <= t) {
      counter <- counter + 1
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      tmp <- biomarkers_t[,c(1, s+1, t+2)]
      out <- maxTPR(tmp, 0.1, approxh=h_sa[counter])
      b1_sTPF <- out$sTPRrslt[3]
      b2_sTPF <- out$sTPRrslt[4]
      v1 <- b1_sTPF*m11 + b2_sTPF*m21
      v0 <- b1_sTPF*m10 + b2_sTPF*m20
      TPFe_sTPF <- c(TPFe_sTPF, eTPF(0.1, v0, v1))
    }
  }
}
TPFe_sTPF

## optimization toward a smooth approximation to pAUC(0, fpf)
TPFe_spAUC <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for  (t in 1:14) {
    if (s <= t) {
      counter <- counter + 1
      m11 <- case_1[, s]
      m10 <- cont_0[, s]
      m21 <- case_1[, t+1]
      m20 <- cont_0[, t+1]
      tmp <- biomarkers_t[,c(1, s+1, t+2)]
      out <- grid.auc(tmp[,1]~tmp[,2]+tmp[,3], tmp, beta_v, approx.type="norm
al", approx.param=h_sa[counter], lambda=0, t0=0, t1=0.1)
      b1_spAUC <- out$coefficient[1]
      b2_spAUC <- out$coefficient[2]
      v1 <- b1_spAUC*m11 + b2_spAUC*m21
      v0 <- b1_spAUC*m10 + b2_spAUC*m20
      TPFe_spAUC <- c(TPFe_spAUC, eTPF(0.1, v0, v1))
    }
  }
```

```
}
TPFe_spAUC

## implement a 10-fold cross-validation on the optimization toward AUC
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
gamma_angle <- seq(1, 360, by=1)
eauc_tr <- vector("numeric")
AUCecv_do <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1[-flds1[[f]],s]
        m10cv <- cont_0[-flds0[[f]],s]
        m21cv <- case_1[-flds1[[f]],t+1]
        m20cv <- cont_0[-flds0[[f]],t+1]
        for (g in 1:length(gamma_angle)) {
          b1cv <- cos((gamma_angle[g]*pi)/180)
          b2cv <- sin((gamma_angle[g]*pi)/180)
          v1cv <- b1cv*m11cv + b2cv*m21cv
          v0cv <- b1cv*m10cv + b2cv*m20cv
          eauc_tr <- c(eauc_tr, eAUC(v0cv, v1cv))
        }
        max_gamma <- gamma_angle[which.max(eauc_tr)]
        b1f <- cos((max_gamma*pi)/180)
        b2f <- sin((max_gamma*pi)/180)
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
        eauc_tr <- vector("numeric")
      }
      AUCecv_do <- c(AUCecv_do, eAUC(v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
AUCecv_do

## implement a 10-fold cross-validation on the optimization toward logistic likelihood
set.seed(123)
```

```
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
AUCecv_lr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1_lr[-flds1[[f]],s+1]
        m10cv <- cont_0_lr[-flds0[[f]],s+1]
        m21cv <- case_1_lr[-flds1[[f]],t+2]
        m20cv <- cont_0_lr[-flds0[[f]],t+2]
        dat <- as.data.frame(cbind(c(case_1_lr[-flds1[[f]],1], cont_0_lr[-fld
s0[[f]],1]), c(m11cv,m10cv), c(m21cv, m20cv)))
        fit <-  glm(dat[,1] ~ dat[,2] + dat[,3], dat, family="binomial")
        b1f <- fit$coef[2]
        b2f <- fit$coef[3]
        m11f <- case_1_lr[flds1[[f]],s+1]
        m10f <- cont_0_lr[flds0[[f]],s+1]
        m21f <- case_1_lr[flds1[[f]],t+2]
        m20f <- cont_0_lr[flds0[[f]],t+2]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
      }
      AUCecv_lr <- c(AUCecv_lr, eAUC(v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
AUCecv_lr
```

implement a 10-fold cross-validation on the optimization toward $TPF|_{fpf}$

```
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
etpf_tr <- vector("numeric")
TPFecv_do <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1[-flds1[[f]],s]
```

```
        m10cv <- cont_0[-flds0[[f]],s]
        m21cv <- case_1[-flds1[[f]],t+1]
        m20cv <- cont_0[-flds0[[f]],t+1]
        for (g in 1:length(gamma_angle)) {
          b1cv <- cos((gamma_angle[g]*pi)/180)
          b2cv <- sin((gamma_angle[g]*pi)/180)
          v1cv <- b1cv*m11cv + b2cv*m21cv
          v0cv <- b1cv*m10cv + b2cv*m20cv
          etpf_tr <- c(etpf_tr, eTPF(0.1, v0cv, v1cv))
        }
        max_gamma <- gamma_angle[which.max(etpf_tr)]
        b1f <- cos((max_gamma*pi)/180)
        b2f <- sin((max_gamma*pi)/180)
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
        etpf_tr <- vector("numeric")
      }
      TPFecv_do <- c(TPFecv_do, eTPF(0.1, v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
TPFecv_do

## implement a 10-fold cross-validation on the optimization toward pAUC(0, fpf)
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
epauc_tr <- vector("numeric")
TPFecv_pauc <- vector("numeric")
counter <- 0
v1f <- vector("numeric")
v0f <- vector("numeric")
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1[-flds1[[f]],s]
        m10cv <- cont_0[-flds0[[f]],s]
        m21cv <- case_1[-flds1[[f]],t+1]
        m20cv <- cont_0[-flds0[[f]],t+1]
        for (g in 1:length(gamma_angle)) {
          b1cv <- cos((gamma_angle[g]*pi)/180)
          b2cv <- sin((gamma_angle[g]*pi)/180)
```

```
          v1cv <- b1cv*m11cv + b2cv*m21cv
          v0cv <- b1cv*m10cv + b2cv*m20cv
          epauc_tr <- c(epauc_tr, epAUC(fpf0=0.1, v0, v1))
        }
        max_gamma <- gamma_angle[which.max(epauc_tr)]
        b1f <- cos((max_gamma*pi)/180)
        b2f <- sin((max_gamma*pi)/180)
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
        epauc_tr <- vector("numeric")
      }
      TPFecv_pauc <- c(TPFecv_pauc, eTPF(0.1, v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
TPFecv_pauc

## implement a 10-fold cross-validation on the optimization toward the smoothed AUC
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
AUCecv_sAUC <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
case_1_cv <- biomarkers_t[biomarkers_t$case==1, ]
cont_0_cv <- biomarkers_t[biomarkers_t$case==0, ]
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1_cv[-flds1[[f]],s+1]
        m10cv <- cont_0_cv[-flds0[[f]],s+1]
        m21cv <- case_1_cv[-flds1[[f]],t+2]
        m20cv <- cont_0_cv[-flds0[[f]],t+2]
        tmp <- as.data.frame(cbind(c(case_1_cv[-flds1[[f]],1], cont_0_cv[-fld
s0[[f]],1]), c(m11cv,m10cv), c(m21cv, m20cv)))
        dat4anal <- model.frame(tmp[,1]~tmp[,2]+tmp[,3], dat=tmp)
        Y <- dat4anal[,1]
        n1 <- sum(Y==1)
        n2 <- sum(Y==0)
        n <- n1+n2
        markers <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3],dat=tmp)[,-1]
```

```
        num_markers <- ncol(markers)
        X1 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==1,
])[,-1,drop=FALSE]
        X2 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==0,
])[,-1,drop=FALSE]
        fit.rlogit  <- rlogit(tmp[,1]~tmp[,2]+tmp[,3], tmp)
        if (fit.rlogit$convergence) {
          beta.init <- fit.rlogit$coef[-1]
        } else {
          beta.init <- rep(1, ncol(tmp)-1)
        }
        Xint <-calXdiff(markers, Y, d=length(beta.init))
        x_diff <- Xint[[1]]
        tu <-  n^(-1/2)*sd(drop(x_diff%*%normsq(beta.init)))
        out_tr <- grid.auc(tmp[,1]~tmp[,2]+tmp[,3], tmp, beta_v, approx.type=
"normal", approx.param=tu, lambda=0)
        b1f <- out_tr$coefficient[1]
        b2f <- out_tr$coefficient[2]
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
      }
      AUCecv_sAUC <- c(AUCecv_sAUC, eAUC(v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
AUCecv_sAUC

## implement a 10-fold cross-validation on the optimization toward STPF|stpf
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
TPFecv_sTPF <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1_cv[-flds1[[f]],s+1]
        m10cv <- cont_0_cv[-flds0[[f]],s+1]
        m21cv <- case_1_cv[-flds1[[f]],t+2]
        m20cv <- cont_0_cv[-flds0[[f]],t+2]
```

```
        tmp <- as.data.frame(cbind(c(case_1_cv[-flds1[[f]],1], cont_0_cv[-fld
s0[[f]],1]), c(m11cv,m10cv), c(m21cv, m20cv)))
        dat4anal <- model.frame(tmp[,1]~tmp[,2]+tmp[,3], dat=tmp)
        Y <- dat4anal[,1]
        n1 <- sum(Y==1)
        n2 <- sum(Y==0)
        n <- n1+n2
        markers <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3],dat=tmp)[,-1]
        num_markers <- ncol(markers)
        X1 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==1,
])[,-1,drop=FALSE]
        X2 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==0,
])[,-1,drop=FALSE]
        fit.rlogit  <- rlogit(tmp[,1]~tmp[,2]+tmp[,3], tmp)
        if (fit.rlogit$convergence) {
          beta.init <- fit.rlogit$coef[-1]
        } else {
          beta.init <- rep(1, ncol(tmp)-1)
        }
        Xint <-calXdiff(markers, Y, d=length(beta.init))
        x_diff <- Xint[[1]]
        tu <-  n^(-1/2)*sd(drop(x_diff%*%normsq(beta.init)))
        out_tr <- maxTPR(tmp, 0.1, approxh=tu)
        if (out_tr$sTPRrslt[5] == 0) {
          out_tr <- maxTPR(tmp, 0.1, approxh=0.5)
        }
        b1f <- out_tr$sTPRrslt[3]
        b2f <- out_tr$sTPRrslt[4]
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
      }
      TPFecv_sTPF <- c(TPFecv_sTPF, eTPF(0.1, v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
TPFecv_sTPF

## implement a 10-fold cross-validation on the optimization toward smoothed pAUC(0, fpf)
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
TPFecv_SpAUC <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
```

```
counter <- 0
for (s in 1:14) {
  for (t in 1:14) {
    if (s <= t) {
      counter <- counter+1
      for (f in 1:10) {
        m11cv <- case_1_cv[-flds1[[f]],s+1]
        m10cv <- cont_0_cv[-flds0[[f]],s+1]
        m21cv <- case_1_cv[-flds1[[f]],t+2]
        m20cv <- cont_0_cv[-flds0[[f]],t+2]
        tmp <- as.data.frame(cbind(c(case_1_cv[-flds1[[f]],1], cont_0_cv[-fld
s0[[f]],1]), c(m11cv,m10cv), c(m21cv, m20cv)))
        dat4anal <- model.frame(tmp[,1]~tmp[,2]+tmp[,3], dat=tmp)
        Y <- dat4anal[,1]
        n1 <- sum(Y==1)
        n2 <- sum(Y==0)
        n <- n1+n2
        markers <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3],dat=tmp)[,-1]
        num_markers <- ncol(markers)
        X1 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==1,
])[,-1,drop=FALSE]
        X2 <- model.matrix(tmp[,1]~tmp[,2]+tmp[,3], dat4anal[dat4anal[,1]==0,
])[,-1,drop=FALSE]
        fit.rlogit  <- rlogit(tmp[,1]~tmp[,2]+tmp[,3], tmp)
        if (fit.rlogit$convergence) {
          beta.init <- fit.rlogit$coef[-1]
        } else {
          beta.init <- rep(1, ncol(tmp)-1)
        }
        Xint <-calXdiff(markers, Y, d=length(beta.init))
        x_diff <- Xint[[1]]
        tu <-  n^(-1/2)*sd(drop(x_diff%*%normsq(beta.init)))
        out_tr <- grid.auc(tmp[,1]~tmp[,2]+tmp[,3], tmp, beta_v, approx.type=
"normal", approx.param=tu, lambda=0, t0=0, t1=0.1)
        b1f <- out_tr$coefficient[1]
        b2f <- out_tr$coefficient[2]
        m11f <- case_1[flds1[[f]],s]
        m10f <- cont_0[flds0[[f]],s]
        m21f <- case_1[flds1[[f]],t+1]
        m20f <- cont_0[flds0[[f]],t+1]
        v1f <- c(v1f, b1f*m11f + b2f*m21f)
        v0f <- c(v0f, b1f*m10f + b2f*m20f)
      }
      TPFecv_SpAUC <- c(TPFecv_SpAUC, eTPF(0.1, v0f, v1f))
      v1f <- vector("numeric")
      v0f <- vector("numeric")
    }
  }
}
TPFecv_SpAUC
```

Appendix B.3 Sequential Optimization of Multiple Biomarkers

```
## sequential AUC
AUCe_do[which.max(AUCe_do)]
max.gamma[which.max(AUCe_do)]
v2 <- biomarkers[,c(1,13,15)]
m11 <- v2[v2$case==1, 2]
m10 <- v2[v2$case==0, 2]
m21 <- v2[v2$case==1, 3]
m20 <- v2[v2$case==0, 3]
b1 <- cos((max.gamma[101]*pi)/180)
b2 <- sin((max.gamma[101]*pi)/180)
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eTPF(0.1, v0, v1)
markers.lm <- data.frame(case=biomarkers_t$case, new.marker=c(v1,v0))
markers_left <- biomarkers_t[,-c(13,15)]
markers_left <- data.frame(markers_left, new.marker=markers.lm$new.marker)
m11 <- markers_left[markers_left$case==1, ncol(markers_left)]
m10 <- markers_left[markers_left$case==0, ncol(markers_left)]
eauc2 <- vector("numeric")
max.gamma <- vector("numeric")
max.eauc <- vector("numeric")
for (sw in 1:(ncol(markers_left)-2)) {
  m21 <- markers_left[markers_left$case==1, sw+1]
  m20 <- markers_left[markers_left$case==0, sw+1]
  for (g in 1:length(gamma_angle)) {
    b1 <- cos((gamma_angle[g]*pi)/180)
    b2 <- sin((gamma_angle[g]*pi)/180)
    v1 <- b1*m11 + b2*m21
    v0 <- b1*m10 + b2*m20
    eauc2 <- c(eauc2, eAUC(v0, v1))
  }
  max.eauc <- c(max.eauc, eauc2[which.max(eauc2)])
  max.gamma <- c(max.gamma, gamma_angle[which.max(eauc2)])
  eauc2 <- vector("numeric")
}
max.eauc[which.max(max.eauc)]
m11 <- markers_left[markers_left$case==1, ncol(markers_left)]
m10 <- markers_left[markers_left$case==0, ncol(markers_left)]
m21 <- markers_left[markers_left$case==1, which.max(max.eauc)+1]
m20 <- markers_left[markers_left$case==0, which.max(max.eauc)+1]
b1 <- cos((max.gamma[which.max(max.eauc)]*pi)/180)
b2 <- sin((max.gamma[which.max(max.eauc)]*pi)/180)
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
eTPF(0.1, v0,v1)
```

```
markers.lm <- data.frame(case=biomarkers_t$case, new.marker=c(v1,v0))
markers_left <- markers_left[,-c(which.max(max.eauc)+1,ncol(markers_left))]
markers_left <- data.frame(markers_left, new.marker=markers.lm$new.marker)

## sequential TPF
m11 <- case_1[, 12]
m10 <- cont_0[, 12]
m21 <- case_1[, 14]
m20 <- cont_0[, 14]
b1 <- cos((maxgamma_etpf[which.max(TPFe_do)]*pi)/180)
b2 <- sin((maxgamma_etpf[which.max(TPFe_do)]*pi)/180)
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eTPF(0.1, v0, v1)
eAUC(v0,v1)
biomarkers_tpf <- biomarkers_t[,-c(13,15)]
biomarkers_tpf <- data.frame(biomarkers_tpf, new.marker=b1*biomarkers_t$v653+
b2*biomarkers_t$v831)
etpf2 <- vector("numeric")
tpf_sw <- vector("numeric")
maxgamma_etpf <- vector("numeric")
m11 <- biomarkers_tpf[biomarkers_tpf$case==1, ncol(biomarkers_tpf)]
m10 <- biomarkers_tpf[biomarkers_tpf$case==0, ncol(biomarkers_tpf)]
for (sw in  1:(ncol(biomarkers_tpf)-2)) {
  m21 <- biomarkers_tpf[biomarkers_tpf$case==1, sw+1]
  m20 <- biomarkers_tpf[biomarkers_tpf$case==0, sw+1]
  for (g in 1:length(gamma_angle)) {
    b1 <- cos((gamma_angle[g]**pi)/180)
    b2 <- sin((gamma_angle[g]**pi)/180)
    v1 <- b1*m11 + b2*m21
    v0 <- b1*m10 + b2*m20
    etpf2 <- c(etpf2, eTPF(0.1, v0, v1))
  }
  tpf_sw <- c(tpf_sw, etpf2[which.max(etpf2)])
  maxgamma_etpf <- c(maxgamma_etpf, gamma[which.max(etpf2)])
  etpf2 <- vector("numeric")
}
maxgamma_etpf[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
tpf_sw[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
b1 <- cos((358*pi)/180)
b2 <- sin((358*pi)/180)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 6+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 6+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.87
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 7+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 7+1]
v1 <- b1*m11 + b2*m21
```

```
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.88
biomarkers_tpf <- data.frame(biomarkers_tpf, new.marker3=b1*biomarkers_tpf$ne
w.marker+b2*biomarkers_tpf$v427)
biomarkers_tpf <- biomarkers_tpf[,-c(8,15)]
etpf2 <- vector("numeric")
tpf_sw <- vector("numeric")
maxgamma_etpf <- vector("numeric")
m11 <- biomarkers_tpf[biomarkers_tpf$case==1, ncol(biomarkers_tpf)]
m10 <- biomarkers_tpf[biomarkers_tpf$case==0, ncol(biomarkers_tpf)]
for (sw in  1:(ncol(biomarkers_tpf)-2)) {
  m21 <- biomarkers_tpf[biomarkers_tpf$case==1, sw+1]
  m20 <- biomarkers_tpf[biomarkers_tpf$case==0, sw+1]
  for (g in 1:length(gamma_angle)) {
    b1 <- cos((gamma_angle[g]**pi)/180)
    b2 <- sin((gamma_angle[g]**pi)/180)
    v1 <- b1*m11 + b2*m21
    v0 <- b1*m10 + b2*m20
    etpf2 <- c(etpf2, eTPF(0.1, v0, v1))
  }
  tpf_sw <- c(tpf_sw, etpf2[which.max(etpf2)])
  maxgamma_etpf <- c(maxgamma_etpf, gamma_angle[which.max(etpf2)])
  etpf2 <- vector("numeric")
}
maxgamma_etpf[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
tpf_sw[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
b1 <- cos((1*pi)/180)
b2 <- sin((1*pi)/180)
biomarkers_tpf <- data.frame(biomarkers_tpf, new.marker4=b1*biomarkers_tpf$ne
w.marker3+b2*biomarkers_tpf$v30)
biomarkers_tpf <- biomarkers_tpf[,-c(2,14)]
etpf2 <- vector("numeric")
tpf_sw <- vector("numeric")
maxgamma_etpf <- vector("numeric")
m11 <- biomarkers_tpf[biomarkers_tpf$case==1, ncol(biomarkers_tpf)]
m10 <- biomarkers_tpf[biomarkers_tpf$case==0, ncol(biomarkers_tpf)]
for (sw in  1:(ncol(biomarkers_tpf)-2)) {
  m21 <- biomarkers_tpf[biomarkers_tpf$case==1, sw+1]
  m20 <- biomarkers_tpf[biomarkers_tpf$case==0, sw+1]
  for (g in 1:length(gamma_angle)) {
    b1 <- cos((gamma_angle[g]**pi)/180)
    b2 <- sin((gamma_angle[g]**pi)/180)
    v1 <- b1*m11 + b2*m21
    v0 <- b1*m10 + b2*m20
    etpf2 <- c(etpf2, eTPF(0.1, v0, v1))
  }
  tpf_sw <- c(tpf_sw, etpf2[which.max(etpf2)])
  maxgamma_etpf <- c(maxgamma_etpf, gamma_angle[which.max(etpf2)])
  etpf2 <- vector("numeric")
```

```
}
maxgamma_etpf[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
tpf_sw[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
b1 <- cos((1*pi)/180)
b2 <- sin((1*pi)/180)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 5+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 5+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.9005
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 8+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 8+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.9007
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 9+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 9+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.9066
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 10+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 10+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.9063
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 11+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 11+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
#0.9061
biomarkers_tpf <- data.frame(biomarkers_tpf, new.marker5=b1*biomarkers_tpf$ne
w.marker4+b2*biomarkers_tpf$v652)
biomarkers_tpf <- biomarkers_tpf[,-c(10,13)
etpf2 <- vector("numeric")
tpf_sw <- vector("numeric")
maxgamma_etpf <- vector("numeric")
m11 <- biomarkers_tpf[biomarkers_tpf$case==1, ncol(biomarkers_tpf)]
m10 <- biomarkers_tpf[biomarkers_tpf$case==0, ncol(biomarkers_tpf)]
for (sw in  1:(ncol(biomarkers_tpf)-2)) {
  m21 <- biomarkers_tpf[biomarkers_tpf$case==1, sw+1]
  m20 <- biomarkers_tpf[biomarkers_tpf$case==0, sw+1]
  for (g in 1:length(gamma_angle)) {
    b1 <- cos((gamma_angle[g]**pi)/180)
    b2 <- sin((gamma_angle[g]**pi)/180)
```

```
    v1 <- b1*m11 + b2*m21
    v0 <- b1*m10 + b2*m20
    etpf2 <- c(etpf2, eTPF(0.1, v0, v1))
  }
  tpf_sw <- c(tpf_sw, etpf2[which.max(etpf2)])
  maxgamma_etpf <- c(maxgamma_etpf, gamma_angle[which.max(etpf2)])
  etpf2 <- vector("numeric")
}
maxgamma_etpf[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
tpf_sw[which(tpf_sw == tpf_sw[which.max(tpf_sw)])]
b1 <- cos((360*pi)/180)
b2 <- sin((360*pi)/180)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 1+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 1+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 2+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 2+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 3+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 3+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 4+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 4+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)
m21 <- biomarkers_tpf[biomarkers_tpf$case==1, 5+1]
m20 <- biomarkers_tpf[biomarkers_tpf$case==0, 5+1]
v1 <- b1*m11 + b2*m21
v0 <- b1*m10 + b2*m20
eAUC(v0,v1)

## implement a 10-fold cross-validation on sequential AUC
markers_opteAUC <- data.frame(case=biomarkers_t$case, v653=biomarkers_t$v653,
                              v831=biomarkers_t$v831, v30=biomarkers_t$v30,
                              v427=biomarkers_t$v427, v652=biomarkers_t$v652,
                              v509=biomarkers_t$v509, v877=biomarkers_t$v877)
case.1.swauc <- markers_opteAUC[markers_opteAUC$case==1,]
cont.0.swauc <- markers_opteAUC[markers_opteAUC$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
eauc_tr <- vector("numeric")
v1f <- vector("numeric")
```

```
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swauc[-flds1[[f]],2]
  m10cv <- cont.0.swauc[-flds0[[f]],2]
  m21cv <- case.1.swauc[-flds1[[f]],3]
  m20cv <- cont.0.swauc[-flds0[[f]],3]
  for (g in 1:length(gamma_angle)) {
    b1cv <- cos((gamma_angle[g]*pi)/180)
    b2cv <- sin((gamma_angle[g]**pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    eauc_tr <- c(eauc_tr, eAUC(v0cv, v1cv))
  }
  max_gamma <- gamma_angle[which.max(eauc_tr)]
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swauc[flds1[[f]],2]
  m10cv <- cont.0.swauc[flds0[[f]],2]
  m21cv <- case.1.swauc[flds1[[f]],3]
  m20cv <- cont.0.swauc[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  eauc_tr <- vector("numeric")
}
eAUC(v0f, v1f)
#0.8767
markers.left.cv <- data.frame(case=markers_opteAUC$case,
                              new.marker_cv=c(v1f,v0f),
                              markers_opteAUC[,c(4:8)])
case.1.swauc <- markers.left.cv[markers.left.cv$case==1,]
cont.0.swauc <- markers.left.cv[markers.left.cv$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
eauc_tr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swauc[-flds1[[f]],2]
  m10cv <- cont.0.swauc[-flds0[[f]],2]
  m21cv <- case.1.swauc[-flds1[[f]],3]
  m20cv <- cont.0.swauc[-flds0[[f]],3]
  for (g in 1:length(gamma_angle)) {
    b1cv <- cos((gamma_angle[g]*pi)/180)
    b2cv <- sin((gamma_angle[g]**pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    eauc_tr <- c(eauc_tr, eAUC(v0cv, v1cv))
  }
  max_gamma <- gamma_angle[which.max(eauc_tr)]
```

```
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swauc[flds1[[f]],2]
  m10cv <- cont.0.swauc[flds0[[f]],2]
  m21cv <- case.1.swauc[flds1[[f]],3]
  m20cv <- cont.0.swauc[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  eauc_tr <- vector("numeric")
}
eAUC(v0f, v1f)
#0.8767
markers.left.cv <- data.frame(case=markers.left.cv$case,
                              new.marker_cv=c(v1f,v0f),
                              markers.left.cv[,c(4:(ncol(markers.left.cv)))])
case.1.swauc <- markers.left.cv[markers.left.cv$case==1,]
cont.0.swauc <- markers.left.cv[markers.left.cv$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
eauc_tr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swauc[-flds1[[f]],2]
  m10cv <- cont.0.swauc[-flds0[[f]],2]
  m21cv <- case.1.swauc[-flds1[[f]],3]
  m20cv <- cont.0.swauc[-flds0[[f]],3]
  for (g in 1:length(gamma_angle)) {
    b1cv <- cos((gamma_angle[g]*pi)/180)
    b2cv <- sin((gamma_angle[g]**pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    eauc_tr <- c(eauc_tr, eAUC(v0cv, v1cv))
  }
  max_gamma <- gamma_angle[which.max(eauc_tr)]
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swauc[flds1[[f]],2]
  m10cv <- cont.0.swauc[flds0[[f]],2]
  m21cv <- case.1.swauc[flds1[[f]],3]
  m20cv <- cont.0.swauc[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  eauc_tr <- vector("numeric")
}
eAUC(v0f, v1f)
eTPF(0.1, v0f, v1f)
markers.left.cv <- data.frame(case=markers.left.cv$case,
                              new.marker_cv=c(v1f,v0f),
```

```
                                v877=markers.left.cv$v877)
case.1.swauc <- markers.left.cv[markers.left.cv$case==1,]
cont.0.swauc <- markers.left.cv[markers.left.cv$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
eauc_tr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swauc[-flds1[[f]],2]
  m10cv <- cont.0.swauc[-flds0[[f]],2]
  m21cv <- case.1.swauc[-flds1[[f]],3]
  m20cv <- cont.0.swauc[-flds0[[f]],3]
  for (h in 1:length(gamma_angle)) {
    b1cv <- cos((h*pi)/180)
    b2cv <- sin((h*pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    eauc_tr <- c(eauc_tr, eAUC(v0cv, v1cv))
  }
  max_gamma <- which.max(eauc_tr)
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swauc[flds1[[f]],2]
  m10cv <- cont.0.swauc[flds0[[f]],2]
  m21cv <- case.1.swauc[flds1[[f]],3]
  m20cv <- cont.0.swauc[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  eauc_tr <- vector("numeric")
}
eAUC(v0f, v1f)
#0.8674
eTPF(0.1, v0f, v1f)
#0.7305

## implement a 10-fold cross-validation on logistic regression (sequentially)
markers_pvals <- data.frame(case=biomarkers_t$case, v877=biomarkers_t$v877,
                            v509=biomarkers_t$v509, v354=biomarkers_t$v354,
                            v30=biomarkers_t$v30, v93=biomarkers_t$v93,
                            v831=biomarkers_t$v831, v426=biomarkers_t$v426,
                            v741=biomarkers_t$v741, v637=biomarkers_t$v637,
                            v365=biomarkers_t$v365, v427=biomarkers_t$v427)
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
case.1.pvals <- markers_pvals[markers_pvals$case==1,]
cont.0.pvals <- markers_pvals[markers_pvals$case==0,]
v1f <- vector("numeric")
```

```
v0f <- vector("numeric")
for (f in 1:10){
  dat4pvals <- as.data.frame(rbind(case.1.pvals[-flds1[[f]], c(1,2,3)], cont.
0.pvals[-flds0[[f]], c(1,2,3)]))
  colnames(dat4pvals) <- c("case", "marker1", "marker2")
  fit <- glm(case~., dat4pvals, family="binomial")
  b1 <- fit$coef[2]
  b2 <- fit$coef[3]
  v1f <- c(v1f, b1*case.1.pvals[flds1[[f]], 2]+b2*case.1.pvals[flds1[[f]], 3]
)
  v0f <- c(v0f, b1*cont.0.pvals[flds0[[f]], 2]+b2*cont.0.pvals[flds0[[f]], 3]
)
}
eAUC(v0f,v1f)
eTPF(0.1,v0f,v1f)
markers_pvals <- markers_pvals[, -c(2,3)]
head(markers_pvals)
markers_pvals <- data.frame(new.marker=c(v1f,v0f), markers_pvals)
markers_pvals <- markers_pvals[,c(2,1,3:(ncol(markers_pvals)))]

## implement a 10-fold cross-validation on sequential TPF
markers_opteTPF <- data.frame(case=biomarkers_t$case, v653=biomarkers_t$v653,
                              v831=biomarkers_t$v831, v427=biomarkers_t$v427,
                              v30=biomarkers_t$v30, v652=biomarkers_t$v652)
case.1.swtpf <- markers_opteTPF[markers_opteTPF$case==1,]
cont.0.swtpf <- markers_opteTPF[markers_opteTPF$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
etpf_tr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swtpf[-flds1[[f]],2]
  m10cv <- cont.0.swtpf[-flds0[[f]],2]
  m21cv <- case.1.swtpf[-flds1[[f]],3]
  m20cv <- cont.0.swtpf[-flds0[[f]],3]
  for (h in 1:length(gamma_angle)) {
    b1cv <- cos((h*pi)/180)
    b2cv <- sin((h*pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    etpf_tr <- c(etpf_tr, eTPF(0.1, v0cv, v1cv))
  }
  max_gamma <- which.max(etpf_tr)
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swtpf[flds1[[f]],2]
  m10cv <- cont.0.swtpf[flds0[[f]],2]
  m21cv <- case.1.swtpf[flds1[[f]],3]
```

```
  m20cv <- cont.0.swtpf[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  etpf_tr <- vector("numeric")
}
eAUC(v0f, v1f)
eTPF(0.1, v0f, v1f)
markers.etpf.cv <- data.frame(case=markers_opteTPF$case,
                              new.marker_cv=c(v1f,v0f),
                              markers_opteTPF[,c(4:6)])
case.1.swtpf <- markers_opteTPF[markers_opteTPF$case==1,]
cont.0.swtpf <- markers_opteTPF[markers_opteTPF$case==0,]
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
etpf_tr <- vector("numeric")
v1f <- vector("numeric")
v0f <- vector("numeric")
for (f in 1:10) {
  m11cv <- case.1.swtpf[-flds1[[f]],2]
  m10cv <- cont.0.swtpf[-flds0[[f]],2]
  m21cv <- case.1.swtpf[-flds1[[f]],3]
  m20cv <- cont.0.swtpf[-flds0[[f]],3]
  for (h in 1:length(gamma_angle)) {
    b1cv <- cos((h*pi)/180)
    b2cv <- sin((h*pi)/180)
    v1cv <- b1cv*m11cv + b2cv*m21cv
    v0cv <- b1cv*m10cv + b2cv*m20cv
    etpf_tr <- c(etpf_tr, eTPF(0.1,v0cv, v1cv))
  }
  max_gamma <- which.max(etpf_tr)
  b1f <- cos((max_gamma*pi)/180)
  b2f <- sin((max_gamma*pi)/180)
  m11cv <- case.1.swtpf[flds1[[f]],2]
  m10cv <- cont.0.swtpf[flds0[[f]],2]
  m21cv <- case.1.swtpf[flds1[[f]],3]
  m20cv <- cont.0.swtpf[flds0[[f]],3]
  v1f <- c(v1f, b1f*m11f + b2f*m21f)
  v0f <- c(v0f, b1f*m10f + b2f*m20f)
  etpf_tr <- vector("numeric")
}
eAUC(v0f, v1f)
eTPF(0.1, v0f, v1f)
markers.etpf.cv <- data.frame(case=markers.etpf.cv$case,
                              new.marker_cv=c(v1f,v0f),
                              markers.etpf.cv[,c(4:(ncol(markers.etpf.cv)))])
markers.etpf.cv <- data.frame(case=markers.etpf.cv$case,
                              new.marker_cv=c(v1f,v0f),
                              v652=markers.etpf.cv$v652)
```

```
markers.etpf.cv <- data.frame(case=markers.etpf.cv$case,
                              new.marker_cv=c(v1f,v0f))
```

Appendix B.4 Simultaneous Optimization of Multiple Biomarkers

```
## lasso
set.seed(123)
training.samples <- biomarkers_t$case %>% createDataPartition(p = 0.8, list =
 FALSE)
train.data  <- biomarkers_t[training.samples, ]
test.data <- biomarkers_t[-training.samples, ]
x <- model.matrix(case~., train.data)[,-1]
x.full <- model.matrix(case~., prostate_lasso)[,-1]
y.full <- prostate_lasso$case
y <- train.data$case
cv.lasso <- cv.glmnet(x, y, alpha = 1, family = "binomial")
full.model.lasso  <- glmnet(x.full, y.full, alpha=1, lambda = cv.lasso$lambda
.min)
betas.lasso <- as.numeric(coef(full.model.lasso))[-1]
new.marker.lasso <- vector("numeric", length=248)
for (i in 1:length(betas.lasso)) {
  new.marker.lasso <- new.marker.lasso+betas.lasso[i]*prostate_lasso[,i+1]
}
v1.lasso <- new.marker.lasso[c(1:167)]
v0.lasso <- new.marker.lasso[c(168:248)]
eAUC(v0.lasso, v1.lasso)
#0.85
eTPF(0.1, v0.lasso, v1.lasso)
#0.58

## ridge regression
set.seed(123)
training.samples <- biomarkers_t$case %>% createDataPartition(p = 0.8, list =
 FALSE)
train.data  <- biomarkers_t[training.samples, ]
test.data <- biomarkers_t[-training.samples, ]
x <- model.matrix(case~., train.data)[,-1]
y <- train.data$case
cv.ridge <- cv.glmnet(x, y, alpha = 0, family = "binomial")
full.model.ridge <- glmnet(x.full, y.full, alpha=0, lambda=cv.ridge$lambda.mi
n)
betas.ridge  <- as.numeric(coef(full.model.ridge))[-1]
new.marker.ridge <- vector("numeric", length=248)
for (i in 1:length(betas.ridge)){
  new.marker.ridge <- new.marker.ridge + (betas.ridge[i]*prostate_ridge[,i+1]
```

```
)
v1.ridge <- new.marker.ridge[c(1:167)]
v0.ridge <- new.marker.ridge[c(168:248)]
eAUC(v0.ridge, v1.ridge)
#0.88
eTPF(0.1, v0.ridge, v1.ridge)
#0.69

## maxTPR
fit.rlogit <- rlogit(case~., biomarkers_t)
fit.rlogit$convergence
beta.init <- fit.rlogit$coef[-1]
n1 <- 167
n2 <- 81
n <- 248
predictors <- model.matrix(case~., biomarkers_t)[,-1]
Y <- biomarkers_t$case
Xint <- calXdiff(predictors, Y, d=length(beta.init))
x_diff <- Xint[[1]]
h15 <- n^(-1/2)*sd(drop(x_diff%*%normsq(beta.init)))
fit.maxTPR <- maxTPR(data= biomarkers_t, tval=0.1, approxh=h15)
betas.maxTPR <- as.numeric(fit.maxTPR$sTPRrslt[3:17])
new.marker.maxTPR <- vector("numeric", length=248)
for (i in 1:length(betas.maxTPR)){
  new.marker.maxTPR <- new.marker.maxTPR + (betas.maxTPR[i]*prostate.maxTPR[,
i+1])
}
v1.maxTPR <- new.marker.maxTPR[c(1:167)]
v0.maxTPR <- new.marker.maxTPR[c(168:248)]
eTPF(0.1, v0.maxTPR, v1.maxTPR)
#0.88
eAUC(v0.maxTPR, v1.maxTPR)
#0.92

## SAUC
fit.nr <- sauc.phi(case~., dat=biomarkers_t)
betas.nr <- fit.nr$coefficients
new.marker.nr <- vector("numeric", length=248)
for (i in 1:length(betas.nr)){
  new.marker.nr <- new.marker.nr + (betas.nr[i]*prostate.nr[,i+1])
}
v1.nr <- new.marker.nr[c(1:167)]
v0.nr <- new.marker.nr[c(168:248)]
eAUC(v0.nr, v1.nr)
#0.91
eTPF(0.1, v0.nr, v1.nr)
#0.8084

## implement a 10-fold cross-validation on lasso
set.seed(123)
```

```
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
new.marker.lasso  <- vector("numeric")
etpf.lasso.f <- vector("numeric")
eauc.lasso.f <- vector("numeric")
for (f in 1:10){
  dat4anal.tr <- as.data.frame(rbind(case_1_cv[-flds1[[f]], ],
                                     cont_0_cv[-flds0[[f]], ]))
  x.tr <- model.matrix(case~., dat4anal.tr)[,-1]
  y.tr <- dat4anal.tr$case
  lasso.cv <- cv.glmnet(x.tr, y.tr, alpha = 1, family = "binomial")
  dat4anal.te <- as.data.frame(rbind(case_1_cv[flds1[[f]], ],
                                     cont_0_cv[flds0[[f]], ]))
  lasso.tr <- glmnet(x.tr, y.tr, alpha=1, lambda=lasso.cv$lambda.min)
  betas.lasso.cv <- as.numeric(coef(lasso.tr))[-1]
  new.marker.lasso.cv <- vector("numeric", length=nrow(dat4anal.te))
  for (p in 1:length(betas.lasso.cv)) {
    new.marker.lasso.cv <- new.marker.lasso.cv+betas.lasso.cv[p]*dat4anal.te[
,p+1]
  }
  dat4anal.f <- data.frame(case=dat4anal.te$case, new.marker.f=new.marker.las
so.cv)
  v1.f <- dat4anal.f[dat4anal.f$case==1, -1]
  v0.f <- dat4anal.f[dat4anal.f$case==0, -1]
  etpf.lasso.f <- c(etpf.lasso.f, eTPF(0.1, v0.f, v1.f))
  eauc.lasso.f <- c(eauc.lasso.f, eAUC(v0.f, v1.f))
  new.marker.lasso <- c(new.marker.lasso, new.marker.lasso.cv)
}
mean(etpf.lasso.f)
mean(eauc.lasso.f)
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
case.cv <- vector("numeric")
for (f in 1:10) {
  case.cv <- c(case.cv, c(case_1_lasso[flds1[[f]], 1], cont_0_lasso[flds0[[f]
], 1]))
}
biomarkers.lasso <- data.frame(case=case.cv, marker=new.marker.lasso)
v1.lasso_cv <- biomarkers.lasso[biomarkers.lasso$case==1,2]
v0.lasso_cv <- biomarkers.lasso[biomarkers.lasso$case==0,2]
eAUC(v0.lasso_cv, v1.lasso_cv)
eTPF(0.1, v0.lasso_cv, v1.lasso_cv)

## implement a 10-fold cross-validation on ridge regression
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
new.marker.ridge <- vector("numeric")
etpf.ridge.f <- vector("numeric")
```

```
eauc.ridge.f <- vector("numeric")
for (f in 1:10){
  dat4anal.tr <- as.data.frame(rbind(case_1_cv[-flds1[[f]], ],
                                     cont_0_cv[-flds0[[f]], ]))
  x.tr <- model.matrix(case~., dat4anal.tr)[,-1]
  y.tr <- dat4anal.tr$case
  ridge.cv <- cv.glmnet(x.tr, y.tr, alpha=0, family = "binomial")
  dat4anal.te <- as.data.frame(rbind(case_1_cv[flds1[[f]], ],
                                     cont_0_cv [flds0[[f]], ]))
  ridge.tr <- glmnet(x.tr, y.tr, alpha=0, lambda=ridge.cv$lambda.min)
  betas.ridge.cv <- as.numeric(coef(ridge.tr))[-1]
  new.marker.ridge.cv <- vector("numeric", length=nrow(dat4anal.te))
  for (p in 1:length(betas.ridge.cv)) {
    new.marker.ridge.cv <- new.marker.ridge.cv + betas.ridge.cv [p]*dat4anal.
te[,p+1]
  }
  dat4anal.f <- data.frame(case=dat4anal.te$case, new.marker.f=new.marker.rid
ge.cv)
  v1.f <- dat4anal.f[dat4anal.f$case==1, -1]
  v0.f <- dat4anal.f[dat4anal.f$case==0, -1]
  etpf.ridge.f <- c(etpf.ridge.f, eTPF(0.1, v0.f, v1.f))
  eauc.ridge.f <- c(eauc.ridge.f, eAUC(v0.f, v1.f))
  new.marker.ridge <- c(new.marker.ridge, new.marker.ridge.cv)
}
mean(etpf.ridge.f)
mean(eauc.ridge.f)
biomarkers.ridge <- data.frame(case=case.cv, marker=new.marker.ridge)
head(biomarkers.ridge)
v1.ridge_cv <- biomarkers.ridge[biomarkers.ridge$case==1,2]
v0.ridge_cv <- biomarkers.ridge[biomarkers.ridge$case==0,2]
eAUC(v0.ridge_cv, v1.ridge_cv)
eTPF(0.1, v0.ridge_cv, v1.ridge_cv)

## implement a 10-fold cross-validation on maxTPR
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
new.marker.maxTPR <- vector("numeric")
etpf.maxTPR.f <- vector("numeric")
eauc.maxTPR.f <- vector("numeric")
for (f in 1:10) {
  dat4anal.tr <- as.data.frame(rbind(case_1_cv[-flds1[[f]], ],
                                     cont_0_cv[-flds0[[f]], ]))
  fit.rlogit.tr <- rlogit(case~.,dat4anal.tr)
  if (fit.rlogit.tr$convergence) {
    beta.init.tr <- fit.rlogit.tr$coef[-1]
  } else {
    beta.init.tr <- rep(1, ncol(dat4anal.tr)-1)
  }
  n1 <- nrow(case_1_maxTPR[-flds1[[f]], ])
```

```
  n2 <- nrow(cont_0_maxTPR[-flds0[[f]], ])
  n <- nrow(dat4anal.tr)
  predictors.tr <- model.matrix(case~., dat4anal.tr)[,-1]
  Y.tr <- dat4anal.tr$case
  Xint.tr <- calXdiff(predictors.tr, Y.tr, d=length(beta.init.tr))
  x_diff.tr <- Xint.tr[[1]]
  h.tr <- n^(-1/2)*sd(drop(x_diff.tr%*%normsq(beta.init.tr)))
  fit.tr <- maxTPR(data=dat4anal.tr, tval=0.1, approxh=h.tr)
  betas.maxTPR.cv <- as.numeric(fit.tr$sTPRrslt[3:17])
  dat4anal.te <- as.data.frame(rbind(case_1_cv[flds1[[f]], ],
                                     cont_0_cv[flds0[[f]], ]))
  new.marker.maxTPR.cv <- vector("numeric", length=nrow(dat4anal.te))
  for (p in 1:length(betas.maxTPR.cv)) {
    new.marker.maxTPR.cv <- new.marker.maxTPR.cv+betas.maxTPR.cv[p]*dat4anal.
te[,p+1]
  }
  dat4anal.f <- data.frame(case=dat4anal.te$case, new.marker.f=new.marker.max
TPR.cv)
  v1.f <- dat4anal.f[dat4anal.f$case==1, -1]
  v0.f <- dat4anal.f[dat4anal.f$case==0, -1]
  etpf.maxTPR.f <- c(etpf.maxTPR.f, eTPF(0.1, v0.f, v1.f))
  eauc.maxTPR.f <- c(eauc.maxTPR.f, eAUC(v0.f, v1.f))
  new.marker.maxTPR <- c(new.marker.maxTPR, new.marker.maxTPR.cv)
}
mean(etpf.maxTPR.f)
mean(eauc.maxTPR.f)
biomarkers.maxTPR <- data.frame(case=case.cv, marker=new.marker.maxTPR)
v1.maxTPR_cv <- biomarkers.maxTPR[biomarkers.maxTPR$case==1,2]
v0.maxTPR_cv <- biomarkers.maxTPR[biomarkers.maxTPR$case==0,2]
eAUC(v0.maxTPR_cv, v1.maxTPR_cv)
eTPF(0.1, v0.maxTPR_cv, v1.maxTPR_cv)

## implement a 10-fold cross-validation on SAUC
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(1:81, k=10)
new.marker.nr  <- vector("numeric")
etpf.nr.f <- vector("numeric")
eauc.nr.f <- vector("numeric")
for (f in 1:10) {
  dat4anal.tr <- as.data.frame(rbind(case_1_cv[-flds1[[f]], ],
                                     cont_0_cv[-flds0[[f]], ]))
  fit.tr <- sauc.phi(case~., dat=dat4anal.tr)
  betas.nr.cv <- fit.tr$coefficients
  dat4anal.te <- as.data.frame(rbind(case_1_cv[flds1[[f]], ],
                                     cont_0_cv[flds0[[f]], ]))
  new.marker.nr.cv <- vector("numeric", length=dim(dat4anal.te)[1])
  for (p in 1:length(betas.nr.cv)) {
    new.marker.nr.cv <- new.marker.nr.cv+betas.nr.cv[p]*dat4anal.te[,p+1]
  }
```

```
  dat4anal.f <- data.frame(case=dat4anal.te$case, new.marker.f=new.marker.nr.
cv)
  v1.f <- dat4anal.f[dat4anal.f$case==1, -1]
  v0.f <- dat4anal.f[dat4anal.f$case==0, -1]
  etpf.nr.f <- c(etpf.nr.f, eTPF(0.1, v0.f, v1.f))
  eauc.nr.f <- c(eauc.nr.f, eAUC(v0.f, v1.f))
  new.marker.nr <- c(new.marker.nr, new.marker.nr.cv)
}
mean(etpf.nr.f)
mean(eauc.nr.f)
biomarkers.nr <- data.frame(case=case.cv, marker=new.marker.nr)
v1.nr_cv <- biomarkers.nr[biomarkers.nr$case==1,2]
v0.nr_cv <- biomarkers.nr[biomarkers.nr$case==0,2]
eAUC(v0.nr_cv, v1.nr_cv)
eTPF(0.1, v0.nr_cv, v1.nr_cv)

## random forests
prostate_rf <- biomarkers_t
prostate_rf$case <- factor(prostate_rf$case, levels=c("1","0"))
fit.rf <- AUCRF(case~., data=prostate_rf)
set.seed(123)
fit.rfcv <- AUCRFcv(fit.rf, nCV=10, M=1)
rfcv.prediction <- predict(fit.rfcv$RFopt, type="vote")[,1]
rf.case <- prostate_rf$case
rf_cv.roc <- roc(rfcv.prediction, rf.case)
rfcv.tpr <-  data.frame(tpr=rfcv.roc$tpr, fpr=rfcv.roc$fpr)
rfcv.tpr
auc(rf_cv.roc)

## implement a 10-fold cross-validation on random forests
set.seed(123)
flds1 <- createFolds(1:167, k=10)
flds0 <- createFolds(168:248, k=10)
case.1.rf <- prostate_rf[biomarkers_t$case==1,]
cont.0.rf <- prostate_rf[biomarkers_t$case==0,]
test.prediction <- vector("numeric")
for (f in 1:10) {
  dat4train <- as.data.frame(rbind(case.1.rf[-flds1[[f]],],
                                   cont.0.rf[-flds0[[f]],]))
  dat4test <- as.data.frame(rbind(case.1.rf[flds1[[f]],],
                                  cont.0.rf[flds0[[f]],]))
  train.fit <- randomForest(case~., data=dat4train, importance=TRUE)
  test.prediction <- c(test.prediction,
                       predict(train.fit, newdata=dat4test, type="vote")[,1])
}
prostate.prediction <- vector("numeric", length=248)
for (o in 1:248) {
  prostate.prediction[as.numeric(names(test.prediction)[o])] <- test.predicti
on[o]
}
```

```
names(prostate.prediction) <- as.character(c(1:248))
prostate_rfcv <- roc(prostate.prediction, prostate_rf$case)
auc(prostate_rfcv)
rfcv.tpr <-  data.frame(tpr=prostate_rfcv$tpr, fpr=prostate_rfcv$fpr)
```

Bibliography

Anderson, T.W., Bahadur, R.R. (1962). Classification into Two Multivariate Normal Distributions with Different Covariance Matrices. The Annals of Mathematical Statistics, 33(2):420-431.

Airola, A., Pahikkala, T., Waegeman, W., De Baets, B., & Salakoski, T. (2011). An Experimental Comparison of Cross-Validation Techniques for Estimating the Area Under the ROC Curve. *Computational Statistics & Data Analysis*, *55*(4), 1828-1844.

Bandos AI, Gur D. (2017, Mar 13). Linear Combinations of Diagnostic Markers for A Specific Clinical Application. *IBS ENAR 2017*, Washington, D.C.

Bansal, A., & Pepe, M. S. (2013). When Does Combining Markers Improve Classification Performance and What are Implications for Practice?. *Statistics in medicine*, *32*(11), 1877–1892. https://doi.org/10.1002/sim.5736

DeLong, E. R., DeLong, D. M., & Clarke-Pearson, D. L. (1988). Comparing the Areas under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach. *Biometrics*, *44*(3), 837–845.

Fong, Y., Yin, S., & Huang, Y. (2016). Combining Biomarkers Linearly and Nonlinearly for Classification Using the Area Under the ROC Curve. *Statistics in medicine*, *35*(21), 3792–3809. https://doi.org/10.1002/sim.6956

Huang, X., Qin, G. and Fang, Y. (2011), Optimal Combinations of Diagnostic Tests Based on AUC. Biometrics, 67: 568-576. https://doi.org/10.1111/j.1541-0420.2010.01450.x

Komori, O., Eguchi, S. (2010). A Boosting Method for Maximizing the Partial Area Under the ROC Curve. *BMC Bioinformatics 11, 314 (2010).* https://doi.org/10.1186/1471-2105-11-314

Lehman, C. D., Arao, R. F., Sprague, B. L., Lee, J. M., Buist, D. S., Kerlikowske, K., Henderson, L. M., Onega, T., Tosteson, A. N., Rauscher, G. H., & Miglioretti, D. L. (2017). National Performance Benchmarks for Modern Screening Digital Mammography: Update from the Breast Cancer Surveillance Consortium. *Radiology*, *283*(1), 49–58. https://doi.org/10.1148/radiol.2016161174

Liaw A. and Wiener M. (2002). Classification and Regression by randomForest. R News 2(3), 18--22.

Lin, H. Z., Zhou, L., Peng, H., & Zhou, X. H. (2011). Selection and Combination of Biomarkers Using ROC Method for Disease Classification and Prediction. *Canadian Journal of Statistics-Revue Canadienne De Statistique*, *39*(2), 324-343. doi:10.1002/cjs.10107

Meisner, A., Carone, M., Pepe, M.; and Kerr, K. F. (2017). Combining Biomarkers by Maximizing the True Positive Rate for A Fixed False Positive Rate. *UW Biostatistics Working Paper Series. Working Paper 420.* Retrieved October 1, 2020, from https://biostats.bepress.com/uwbiostat/paper420/

Pan X, Metz CE. The "proper" binormal model: parametric receiver operating characteristic curve estimation with degenerate data. Academic Radiology.1997; 4: 380- 389.

Pepe, M., Cai, T., & Longton, G. (2006). Combining Predictors for Classification Using the Area Under the Receiver Operating Characteristic Curve. *Biometrics, 62*(1), 221-229. Retrieved October 1, 2020, from http://www.jstor.org/stable/3695724.

Pepe, M. S., Kerr, K. F., Longton, G., & Wang, Z. Y. (2013). Testing for Improvement in Prediction Model Performance. *Statistics in Medicine, 32*(9), 1467-1482.

Pepe, M. S., & Thompson, M. L. (2000). Combining Diagnostic Test Results to Increase Accuracy. *Biostatistics (Oxford, England), 1*(2), 123–140. https://doi.org/10.1093/biostatistics/1.2.123

Su, J.Q., & Liu, J. (1993). Linear Combinations of Multiple Diagnostic Markers. *Journal of the American Statistical Association, 88*, 1350-1355.

Wang, Z., & Chang, Y. C. (2011). Marker Selection via Maximizing the Partial Area Under the ROC Curve of Linear Risk Scores. *Biostatistics (Oxford, England), 12*(2), 369–385. https://doi.org/10.1093/biostatistics/kxq052

Yasui, Y., McLerran, D., Adam, B. L., Winget, M., Thornquist, M., & Feng, Z. D. (2003). An Automated Peak Identification/Calibration Procedure for High-Dimensional Protein Measures from Mass Spectrometers. *Journal of Biomedicine and Biotechnology* (4), 242-248.

Zhou, X.H., Obuchowski, N.A., McClish D.K. (2011). Statistical Methods in Diagnostic Medicine. 2nd Edition. *New York: Wiley & Sons Inc*.