Supplementary Methods

Contents

Stock plate layout

a) Concentration distribution (row):

A	$[]_{stock}$
B	$[]_{stock}$/3
C	$[]_{stock}$/9
D	$[]_{stock}$/27
E	$[]_{stock}$/81
F	$[]_{stock}$/2.4E+2
G	$[]_{stock}$/7.3E+2
H	$[]_{stock}$/2.2E+3
I	$[]_{stock}$/6.6E+3
J	$[]_{stock}$/2.0E+4
K	$[]_{stock}$/5.9E+4
L	$[]_{stock}$/1.8E+5
M	$[]_{stock}$/5.3E+5
N	$[]_{stock}$/1.6E+6
O	$[]_{stock}$/4.8E+6
P	$[]_{stock}$/1.4E+7

b) Compound distribution (column):

	Plate 1	Plate 2	Plate 3	Plate 4	Plate 5
1	DMSO	DMSO	DMSO	DMSO	DMSO
2	1	21	41	61	81
3	2	22	42	62	82
4	3	23	43	63	83
5	4	24	44	64	84
6	5	25	45	65	85
7	6	26	46	66	86
8	7	27	47	67	87
9	8	28	48	68	88
10	9	29	49	69	89
11	10	30	50	70	90
12	DMSO	DMSO	DMSO	DMSO	DMSO
13	DMSO	DMSO	DMSO	DMSO	DMSO
14	11	31	51	71	91
15	12	32	52	72	92
16	13	33	53	73	93
17	14	34	54	74	94
18	15	35	55	75	95
19	16	36	56	76	96
20	17	37	57	77	97
21	18	38	58	78	98
22	19	39	59	79	99
23	20	40	60	80	100
24	DMSO	DMSO	DMSO	DMSO	DMSO

Note: $[]_{stock}$ = stock concentration.

Compound list

Cpd#	Name	$[]_{stock}$ (mM)	Major activity
1	105D	10	Microtubule
2	A23187 free acid	10	Calcium regulation
3	Amanitin	1	RNA
4	Actinomycin D	10	RNA
5	ALLN	10	Protein degradation
6	Alsterpaullone	10	Kinase
7	Anisomycin	10	Protein synthesis
8	Brefeldin A	10	Vesicle trafficking
9	8-bromo-cAMP	10	Kinase; PKA
10	Camptothecin	10	Topoisomerase
11	Chelerythrine	10	Kinase; PKC
12	Ciglitazone	10	Nuclear receptor
13	Colchicine	10	Microtubule
14	Cycloheximide	10	Protein synthesis

Cpd#	Name	$[]_{stock}$ (mM)	Major activity
15	Cyclosporin A	10	Calcium regulation
16	Cytochalasin D	10	Actin
17	Deoxymannojirimycin	10	Vesicle trafficking
18	Deoxynorjrimycin	10	Vesicle trafficking
19	Dexamethasone	10	Nuclear receptor
20	Doxorubicin	10	Topoisomerase
21	Emetine-1	10	Protein synthesis
22	Emodin	10	Kinase
23	Etoposide	10	Topoisomerase
24	Exo1	10	Vesicle trafficking
25	11N84	10	Vesicle trafficking / kinase
26	Forskolin	10	Kinase; PKA
27	Genistein	10	Kinase
28	Griseofulvin	10	Microtubule

Cpd#	Name	[]stock (mM)	Major activity
29	H89	10	Kinase
30	Hydroxy urea	10	DNA Replication
31	Ibuprofen	10	Cyclooxygenase
32	Indirubin monoxime	10	Kinase; CDK
33	Indomethacin	10	Cyclooxygenase
34	Jasplakinolide	1	Actin
35	Lactacystin	1	Protein degradation
36	Latrunculin B	10	Actin
37	Mevastatin	10	Cholesterol
38	MG132	10	Protein degradation
39	Monastrol	10	Microtubule
40	Nocodazole-1	10	Microtubule
41	Okadaic acid	0.1	Kinase
42	Olomucine	10	Kinase; CDK
43	PMA	10	Kinase; PKC
44	Podophyllotoxin	10	Microtubule
45	Puromycin	10	Protein synthesis
46	Purvalanol A	10	Kinase; CDK
47	Rapamycin	10	Kinase; PI3K pathway
48	Retinoic acid (trans)	10	Nuclear receptor
49	Roscovitine	10	Kinase; CDK
50	ICRF193	10	Topoisomerase
51	Staurosporine	1	Kinase
52	Sulindac sulfide	10	Cyclooxygenase
53	Taxol	10	Microtubule
54	Trichostatin	10	Histone deacetylase
55	Tunicamycin	6	Vesicle trafficking
56	U0126	10	Kinase; MAPK/ERK pathway
57	Vinblastine	10	Microtubule
58	W-7 hydrochloride	10	Calcium regulation
59	Wortmannin	10	Kinase; PI3K pathway
60	WY-14643	10	Nuclear receptor
61	Cytochalasin B	10	Actin
62	Chloropromazine	10	Neurotransmitter
63	PD98059	10	Kinase; MAPK/ERK pathway
64	Clozapine	10	Neurotransmitter
65	Trifluoperazine	10	Neurotransmitter

Cpd#	Name	[]stock (mM)	Major activity
66	SB202190	10	Kinase; MAPK/p38 pathway
67	LY294002	10	Kinase; PI3K pathway
68	Sodium butyrate	10	Histone deacetylase
69	Nitropropionate	10	Energy metabolism
70	Simavastatin	10	Cholesterol
71	Niflumic acid	10	Cyclooxygenase
72	Fluobiprofen	10	Cyclooxygenase
73	Fluoxetine	10	Neurotransmitter
74	Scriptaid-1	10	Histone deacetylase
75	SC560	10	Cyclooxygenase
76	Apicidin	10	Histone deacetylase
77	Epothilone B	0.1	Microtubule
78	Oxamflatin	10	Histone deacetylase
79	SC236	10	Cyclooxygenase
80	SB203580	10	Kinase; MAPK/p38 pathway
81	Aphidicolin	10	DNA Replication
82	PD169316	10	Kinase; MAPK/p38 pathway
83	Methotrexate	10	DNA Replication
84	Ceramide	10	Kinase; PKC
85	Leupeptine	10	Protein degradation
86	Sodium azide	10	Energy metabolism
87	Zvad	1	Protein degradation
88	CKI7	10	Kinase
89	TPEN	10	Metal homeostasis
90	Oligomycin	10	Energy metabolism
91	Nocodazole-2	33	Microtubule
92	Nocodazole-3	0.67	Microtubule
93	Indomethacin	25	Cyclooxygenase
94	Hydroxy urea-2	197	DNA Replication
95	Filopodine	36	Unknown
96	Emetine-2	50	Protein synthesis
97	Scriptaid-2	10	Histone deacetylase
98	Didemnin B	4.5	Protein synthesis
99	Austocystin	13	Unknown
100	Concentramide	10	Unknown

Feature List

Note: DAPI = DNA marker, W2 = second fluorescence marker, and W3 = third fluorescence marker. Objects are binary masks resulting for a threshold segmentation on the intensity value of a marker. c_a = centroid of fluorescence marker intensities, c_ao = centroid of fluorescent marker objects, and c_d = centroid of DNA intensities. GLCM = Grey Level Co-occurrence Matrix.

No.	*Features*
	Morphology Features
1	Area of DNA region
2	Area of cell region
3	Area of non-DNA region
4	Area of DNA region/area of cell region
5	Perimeter of DNA region
6	Eccentricity of DNA region
7	Shape factor of DNA region
8	Solidity of DNA region
9	W2 number of objects in cell region
10	W3 number of objects in cell region
11	W2 ratio of the largest object to the smallest in cell region
12	W3 ratio of the largest object to the smallest in cell region
13	W2 euler number in cell region
14	W3 euler number in cell region
	Texture Features
15	DAPI Haralick GLCM mean angular second moment
16	DAPI Haralick GLCM mean contrast
17	DAPI Haralick GLCM mean correlation
18	DAPI Haralick GLCM mean variance
19	DAPI Haralick GLCM mean inverse difference moment
20	DAPI Haralick GLCM mean sum average
21	DAPI Haralick GLCM mean sum variance
22	DAPI Haralick GLCM mean sum entropy
23	DAPI Haralick GLCM mean entropy
24	DAPI Haralick GLCM mean difference variance
25	DAPI Haralick GLCM mean difference entropy
26	DAPI Haralick GLCM mean measure of correlation 1
27	DAPI Haralick GLCM mean measure of correlation 2
28	DAPI Haralick GLCM standard deviation angular second moment
29	DAPI Haralick GLCM standard deviation contrast
30	DAPI Haralick GLCM standard deviation correlation
31	DAPI Haralick GLCM standard deviation variance
32	DAPI Haralick GLCM standard deviation inverse difference moment
33	DAPI Haralick GLCM standard deviation sum average
34	DAPI Haralick GLCM standard deviation sum variance
35	DAPI Haralick GLCM standard deviation sum entropy
36	DAPI Haralick GLCM standard deviation entropy
37	DAPI Haralick GLCM standard deviation difference variance
38	DAPI Haralick GLCM standard deviation difference entropy
39	DAPI Haralick GLCM standard deviation measure of correlation 1
40	DAPI Haralick GLCM standard deviation measure of correlation 2
41	W2 Haralick GLCM mean angular second moment
42	W2 Haralick GLCM mean contrast
43	W2 Haralick GLCM mean correlation
44	W2 Haralick GLCM mean variance
45	W2 Haralick GLCM mean inverse difference moment
46	W2 Haralick GLCM mean sum average
47	W2 Haralick GLCM mean sum variance
48	W2 Haralick GLCM mean sum entropy
49	W2 Haralick GLCM mean entropy
50	W2 Haralick GLCM mean difference variance

No.	*Features*
51	W2 Haralick GLCM mean difference entropy
52	W2 Haralick GLCM mean measure of correlation 1
53	W2 Haralick GLCM mean measure of correlation 2
54	W2 Haralick GLCM standard deviation angular second moment
55	W2 Haralick GLCM standard deviation contrast
56	W2 Haralick GLCM standard deviation correlation
57	W2 Haralick GLCM standard deviation variance
58	W2 Haralick GLCM standard deviation inverse difference moment
59	W2 Haralick GLCM standard deviation sum average
60	W2 Haralick GLCM standard deviation sum variance
61	W2 Haralick GLCM standard deviation sum entropy
62	W2 Haralick GLCM standard deviation entropy
63	W2 Haralick GLCM standard deviation difference variance
64	W2 Haralick GLCM standard deviation difference entropy
65	W2 Haralick GLCM standard deviation measure of correlation 1
66	W2 Haralick GLCM standard deviation measure of correlation 2
67	W3 Haralick GLCM mean angular second moment
68	W3 Haralick GLCM mean contrast
69	W3 Haralick GLCM mean correlation
70	W3 Haralick GLCM mean variance
71	W3 Haralick GLCM mean inverse difference moment
72	W3 Haralick GLCM mean sum average
73	W3 Haralick GLCM mean sum variance
74	W3 Haralick GLCM mean sum entropy
75	W3 Haralick GLCM mean entropy
76	W3 Haralick GLCM mean difference variance
77	W3 Haralick GLCM mean difference entropy
78	W3 Haralick GLCM mean measure of correlation 1
79	W3 Haralick GLCM mean measure of correlation 2
80	W3 Haralick GLCM standard deviation angular second moment
81	W3 Haralick GLCM standard deviation contrast
82	W3 Haralick GLCM standard deviation correlation
83	W3 Haralick GLCM standard deviation variance
84	W3 Haralick GLCM standard deviation inverse difference moment
85	W3 Haralick GLCM standard deviation sum average
86	W3 Haralick GLCM standard deviation sum variance
87	W3 Haralick GLCM standard deviation sum entropy
88	W3 Haralick GLCM standard deviation entropy
89	W3 Haralick GLCM standard deviation difference variance
90	W3 Haralick GLCM standard deviation difference entropy
91	W3 Haralick GLCM standard deviation measure of correlation 1
92	W3 Haralick GLCM standard deviation measure of correlation 2
	Moment Features
93	DAPI distance between c_a and c_ao
94	W2 distance between c_a and c_ao
95	W3 distance between c_a and c_ao
96	W2 distance between c_a and c_d
97	W3 distance between c_a and c_d
98	W2 average object distance from the c_d
99	W3 average object distance from the c_d
100	W2 standard deviation of object distance to the c_d
101	W3 standard deviation of object distance to the c_d
102	W2 average object distance to the c_a
103	W3 average object distance to the c_a
104	W2 standard deviation of object distance to the c_a
105	W3 standard deviation of object distance to the c_a
	Zernike Features
106	DAPI Zernike moment 1
…	…
154	DAPI Zernike moment 49
155	W2 Zernike moment 1
…	…
203	W2 Zernike moment 49

No.	*Features*
204	W3 Zernike moment 1
…	…
252	W3 Zernike moment 49
	Intensity Features
253	DAPI total intensity in DNA region
254	W2 total intensity in DNA region
255	W3 total intensity in DNA region
256	W2 total intensity in DNA region/W2 total intensity in non-DNA region
257	W3 total intensity in DNA region/W3 total intensity in non-DNA region
258	W2 average intensity in cell region
259	W3 average intensity in cell region
260	DAPI average intensity in DNA region
261	W2 average intensity in DNA region
262	W3 average intensity in DNA region
263	W2 average intensity in non-DNA region
264	W3 average intensity in non-DNA region
265	W2 average intensity in DNA region/W2 average intensity in non-DNA region
266	W3 average intensity in DNA region/W3 average intensity in non-DNA region
267	W2 average intensity in DNA region/DAPI average intensity in DNA region
268	W3 average intensity in DNA region/DAPI average intensity in DNA region
269	W2 standard deviation intensity in cell region
270	W3 standard deviation intensity in cell region
271	DAPI standard deviation intensity in DNA region
272	W2 standard deviation intensity in DNA region
273	W3 standard deviation intensity in DNA region
274	W2 correlation between antibody and DAPI in cell region
275	W3 correlation between antibody and DAPI in cell region
276	W3 correlation between antibody and W2 in cell region
	Random Features
277	Random features 1
…	…
296	Random features 20

Support vector machine

A support vector machine (SVM)[1] with linear kernel function is used in our study because it performs well in many real classification problems[2, 3]. In general, the decision function of a SVM is given by

$$f(X) = \langle W, \phi(X) \rangle + b$$
$$= \sum_{i=1}^{n} \alpha_i y_i \langle \phi(X_i'), \phi(X) \rangle + b$$

where W is a normal vector to the decision hyperplane, b is a bias term, X is an input sample, X_i' is a training sample, α_i is a coefficient determined by the SVM algorithm, y_i is the class label of the i-th training sample, n is the number of training sample, $\phi(\cdot)$ is a function that maps the input sample to some space, and $\langle , \rangle$ is the dot product operator. The function $K(X_t', X) = \langle \phi(X_i'), \phi(X) \rangle$ is also called a kernel function. In our case, a linear kernel function is $K(X_t', X) = \langle X_i', X \rangle$. Other kernel functions with non-linear mapping functions, such as the polynomial and Gaussian kernel functions, have the potential to separate nonlinearly separable feature values[3]. However, these nonlinear kernel functions are not used in our system due to the following reasons:

1. Non-linear kernel functions map the feature values into a very high dimensional space. So, the normal vector W of the decision hyperplane in the mapped feature space, which is used by us as the compound profile, also has the same high dimension. For example, a d-degree polynomial kernel function map the feature values to the $\binom{d+m-1}{d}$-th dimensional space[3]. For d=3 and m=296, the mapped dimension is around 4.4 million. Given the current limitation of computational hardware, it is unfeasible to store and process these very high dimensional extracted profiles.
2. The mapping functions $\phi(\cdot)$ of some non-linear kernel functions are either unknown or unsolvable. The determination of the normal vector requires the mapping function: $W = \sum_{i=1}^{n} \alpha_i y_i \phi(X_i')$. For some non-linear kernel functions, such as the popular Gaussian kernel function, there is no explicit form of $\phi(\cdot)$[3], in which case the normal vector cannot be determined.

A SVM algorithm with linear kernel function has two main parameters, the penalty parameter (C) for the error term in SVM, and the tolerance of termination criterion (*SVM_ERROR*). *SVM_ERROR* was selected to be 0.001. A higher value was found to cause numerical instability in the SVM algorithm for some data (data not shown). To determine the best value for the penalty parameter, a grid search on the parameters using 2-fold cross validation with 3 randomized fold divisions was performed on a prototype dataset (**Supplementary Methods** online). The values of $C = \{0.01, 0.1, 1, 10\}$ were considered. The estimated classification accuracies (Profile and classification accuracy computation, **Methods** in main text) and

computational runtimes for different values of the parameter were obtained (**Supplementary Table M1a**). Overall, the classification accuracy increased as *C* increased. We found that *C*=1 and 10 gave comparable classification accuracy, while *C*=10 required a significant more computational time than *C*=1. Thus, *C* = 1 was chosen to be the penalty parameter.

Perturbation-based cluster validation algorithm

The optimum number of partitions was determined automatically by choosing the most stable clustering result under random perturbations. An extensive list of references on stability-based clustering validation algorithms can be found in a recent review[4]. The original concept of cluster stability[5, 6] used resampling to perturb the clustering results. An inconsistency value for the clustering results after many trials of perturbation was estimated, and normalized by the expected inconsistency value obtained from random clusterings.

When a dataset has a small number of objects (*e.g.*, 10-13 profiles per compound), the original approach based on resampling produces perturbations with low diversity. To overcome this difficulty, perturbation was introduced by adding randomly generated, normally distributed noise to the each feature. Similar ideas of adding random noise to perturb the data were also used previously[7, 8]. The performance of the clustering validation algorithm depends on the noise level. The noise level cannot be extremely small or large, as it will always produce clustering results with low or high inconsistency ratios. We have found empirically that when the noise level on each feature is equal to the standard deviation of the feature, the clustering validation method produced good results (**Supplementary Notes** online).

The algorithm is described below:

Given a set of possible *k* values and a set of profiles:

1. Add random noise to the original dataset to generate a training dataset.
2. Add random noise to the original dataset to generate a test dataset.
3. Cluster each of the training and test datasets into *k* clusters, and assign a cluster label (from 1 to *k*) to each profile.
4. Train a nearest-neighbor classifier on the training dataset.
5. Predict the cluster label of the test dataset using the trained classifier.
6. Calculate the inconsistency ratio by dividing the number of profiles with different predicted and assigned cluster memberships by the total number of profiles. Since the same cluster may have different predicted and assigned class labels, a Hungarian method[9] is used to find the optimum matching between the two label sets.
7. Repeat step 1-6 for 100 times, and calculate the average inconsistency ratio for *k*.
8. Repeat step 1-6 for 100 times with a random classifier, and calculate the average random inconsistency ratio.
9. Normalize the average inconsistency ratio with the average random inconsistency ratio.
10. Repeat step 1-9 for all *k* values, and select *k* with the minimum average normalized inconsistency ratio.

Dendrogram leaf reordering

We used the MatArray Toolbox[10] to select an optimum reordering of the leaves of a dendrogram to maximize the sum of the similarities of adjacent leaves.

Segmentation of DNA and non-DNA regions

For the purpose of segmentation, the intensity range of an image was first adjusted to 0 and 1 by clipping the top and bottom 1% intensity values. After that, the image was smoothed by convolving it with a 3x3 2D Gaussian lowpass filter ($\sigma = 3$). For images of DNA-SC35-anillin and DNA-MT-actin, a 6x6 2D Gaussian lowpass filer ($\sigma = 6$) was used.

The segmentation process consisted of two major steps. In the first step, the DNA image was segmented to identify the DNA regions (the foreground objects) from the background intensities (the background objects). Binary foreground and background masks were first created. The foreground mask was created using the *h*-dome operator[11] and an edge detector on the DNA image. The height of the *h*-dome was selected to be 0.4 of the difference between the maximum and minimum intensity values of the DNA image. A Laplacian of Gaussian zero-crossing edge detector was used to identified edges of foreground objects at the zero-crossing of the DNA image convolved with a Laplacian of Gaussian filter of width 3 pixels and $\sigma = 3$. The union of the binary objects resulting from the *h*-dome operator and edge detector was used as the foreground mask. A background mask was created using Otsu's segmentation algorithm[12] on the DNA image. In general, the algorithm detected most of the background objects, but produced connecting foreground objects for touching cells. A watershed algorithm was used to break apart these connected cells. The watershed algorithm[13] was performed on the inverted foreground mask. The union of the crest lines determined from the watershed algorithm and the background mask provided the boundaries for the DNA regions.

In the second step, a composite cell image obtained from the linear combination of the images on all three channels was segmented to identify the regions stained with fluorescent markers. Similar foreground and background masks creation procedures and watershed algorithm were performed on the composite cell images, except the union of the binary objects resulting from the *h*-dome operator, edge detector, and the foreground objects from the first step (the DNA region) was used as the foreground mask. After the stained regions were identified, the non-DNA regions were then recovered by eliminating the DNA regions from the stained regions.

Suboptimal algorithm for representative d-profiles selection

For each compound category with n_d>15, we first removed d-profiles with insignificant AUC values ($P > 0.05$). Then, for each compound with more than 1 d-profile, we iteratively retained 1 of the compound's d-profiles, removed the compound's other d-profiles, and recalculated the average *P*-value for AUC for the compound category with the new set of retained d-profiles. The d-profile with the minimum average *P*-value was selected to represent the compound.

Prototype compounds selection

Four different compounds, each on a different compound category and a different marker set (namely Camptothecin on DNA-SC35-anillin, Hydroxy Urea-2 on DNA-p53-cFos, Oxamflatin on DNA-pp38-pERK, and Taxol on DNA-MT-actin) were used as the prototype compounds for parameter tunings and algorithm comparisons. These compounds were selected because they gave clearly detectable responses in a preliminary test of our method (data not shown). Due to the large amount of data and limitation in computational power, performing parameter tunings and algorithm comparisons on the whole dataset was unfeasible. These prototypes were used in **Supplementary Table M1** and **Supplementary Notes** online.

Feature selection algorithm candidates

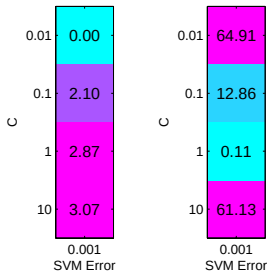
Univariate *t*-Test (TTEST) and Kolmogorov-Smirnov Test (KSTEST). The absolute value of the t-statistic or the Kolmogorov-Smirnov statistic was estimated for each feature. Then, the features were ranked according to the estimated statistics. Starting with the highest ranked feature, the features were iteratively selected in the order of their rankings. The matlab functions TTEST2 or KSTEST2 were used to calculate the statistics.

Multivariate Stepwise Discriminant Analysis (SDA). The multivariate Wilks' lamda statistic was used to measure the separation of treated and control samples on a subset of features[14]. Starting from an empty set, features were iteratively removed or added to decrease the multivariate partial lambda-statistic. The SDA algorithm was implemented as previously described[14]. The tolerance threshold was set to be 0.001. The performance of the algorithm depends on two major parameters, the significance threshold for adding a feature (*F_IN*) and threshold for removing a feature (*F_OUT*). Testing on a prototype dataset (**Supplementary Methods** online) showed that *F_IN*=0.10 and *F_OUT*=0.10 gave the best average classification accuracy (**Supplementary Table M1b**).

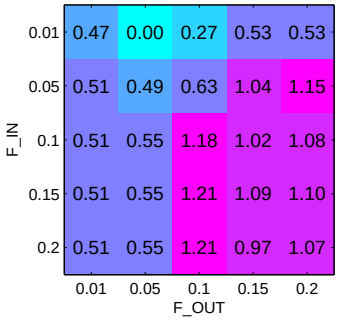
Multivariate Genetic Algorithm (GA). The algorithm searched randomly for the best performing feature subset[15]. The Genetic Algorithm Optimization Toolbox (http://www.ise.ncsu.edu/mirage/GAToolBox/gaot/) was used. The fitness function calculated the SVM classification accuracy. The initial population size was 50 and the maximum number of iteration was 100. Higher population size and number of iteration increases the chance of finding better feature subset, but in practice, they are bounded by the available computational resources. A grid search on a prototype dataset (**Supplementary Methods** online) was used to determine the crossover and mutation rates (**Supplementary Table M1c**). The result showed that average classification accuracy increased with decreasing mutation rate, and reached a maxima around mutation rate = 0.05. The average classification accuracy was not sensitive to the changes in the crossover rate, and crossover rate=0.95 was chosen.

Supplementary Table M1: Parameter tuning for feature selection algorithms

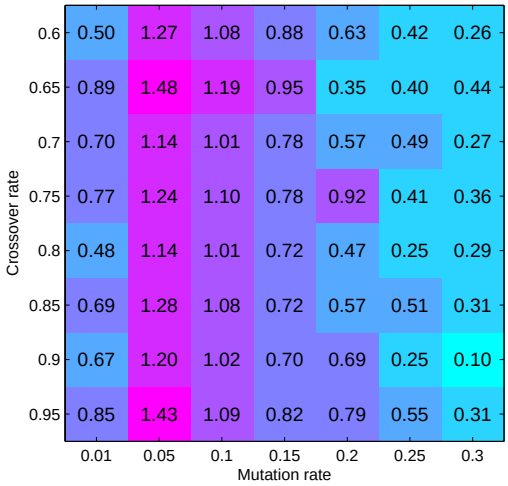
a) Percentage of increase in average classification accuracy (left) and average computational time (right)



b) Percentage of increase in average classification accuracy



c) Percentage of increase in average classification accuracy



Parameters of feature selection algorithms were tuned on the prototype compounds for the highest increase in average classification accuracy. For each prototype, the percentage of increase in average classification accuracy at a parameter value was calculated by dividing the classification accuracy obtained at the parameter value with the minimum classification accuracy achieved over all the parameter values considered, and subtracting one from the result. After that, the percentage of increase in average classification accuracy at a parameter value was averaged across all four prototype compounds. The normalization step was necessary because each prototype compound has a different range of classification accuracy. The percentage of increase in computational runtime was computed similarly. Results for **a)** support vector machine recursive feature elimination algorithm, **b)** stepwise discriminant analysis, and **c)** genetic algorithm. (Pink = higher values, light blue = lower values.)

Percentage of random features selected

Irrelevant features are features that do not provide information about the drug effects of a compound. Let m = number of features, m_s = number of selected features, m_r = number of irrelevant features, m_{sr} = number of selected features that are irrelevant (false positives), the percentage of irrelevant features selected is then given by

$$\frac{m_{sr}}{m_s} = \left(\frac{m_{sr}}{m_r}\right)\frac{\left(\frac{m_r}{m}\right)}{\left(\frac{m_s}{m}\right)} = (\text{false positive rate})\frac{\text{irrelevant feature rate}}{\text{feature selection rate}}.$$

For a given dataset, the irrelevant feature rate is a constant. Thus the percentage of irrelevant features selected is equal to the false positive rate normalized by the feature selection rate. For our application, this criterion is more useful than the false positive rate because a feature selection algorithm that selects a smaller number of features has a higher chance of achieving smaller false positive rate, yet most of the features selected by the algorithm may be irrelevant.

Since the identities of the irrelevant features are unknown in our dataset, the percentage of irrelevant features selected cannot be estimated directly. However, we can add random features, which are artificially generated from noise, to the data and measure the number of random features selected m'_{sr}. Since $\frac{m'_{sr}}{m_s} \le \frac{m_{sr}}{m_s}$, the percentage of random features selected estimates the lower bound of the percentage of irrelevant features selected.

Approximate paired statistical test for comparing feature selection algorithms

A 2x3 cross-validation paired *t*-test was used for comparing the performances of two feature selection algorithms. This statistical test is based on the 5x3 cross-validation paired *t*-test[16], which has the highest power among other statistical tests with acceptable false positive rates for comparing classifiers[16]. In the original proposed approach, a 2-fold cross validation with 5 random fold divisions was proposed (hence the name 5x2). Due the large number of samples (~2400 treated cells and ~2400 control cells were used for each estimation of classification accuracy), each random fold division generated very little variation on the estimated classification accuracy. Thus, 3 random fold divisions were sufficient for our dataset.

For our study, the performances of 6 feature selection algorithms were compared simultaneously to SVMRFE2.. The multiple hypothesis testing was corrected by controlling the false discovery rate[17], instead of using the more conservative Bonferroni correction. A *q*-value threshold (threshold for false discovery rate) of 0.10 was used. All comparisons were one-tailed and the actual null hypotheses tested are given in the figure legends of the results (**Supplementary Fig. N1-5**, **Supplementary Notes** online).

Occasionally, the difference between two feature selection algorithms did not change across all cross-validation folds and random fold divisions. In these cases, the paired *t*-test could not be

used and the algorithms were specially indicated (x in **Supplementary Fig. N1-5**, **Supplementary Notes** online).

Quality control

A semi-automated algorithm was used to identify potentially bad wells. For every plate, the distributions of the number and the area of cells on each well were determined. Wells with abnormal cell number or cell area were identified and manually examined to remove bad images from further analysis. Most of the removed images were either corrupted by imaging artifacts or were empty with no cells. The well-to-well variation of cell numbers on the control wells of every plate was also examined, and the highest and two lowest dilutions (rows A, O and P) were dropped due to persistent low cell numbers, resulting in retaining 13 of the original 16 serial 3-fold dilutions.

References

1. Vapnik, V.N. Statistical learning theory. (Wiley, 1998).
2. Cristianini, N. & Shawe-Taylor, J. An introduction to Support Vector Machines and other kernel-based learning methods. (Cambridge University Press, Cambridge, UK; 2000).
3. Schölkopf, B. & Smola, A.J. Learning with Kernels. (MIT Press, Cambridge, MA; 2001).
4. Handl, J., Knowles, J. & Kell, D.B. Computational cluster validation in post-genomic data analysis. *Bioinformatics* **21**, 3201-3212 (2005).
5. Lange, T., Roth, V., Braun, M.L. & Buhmann, J.M. Stability-Based Validation of Clustering Solutions. *Neural Comp.* **16**, 1299-1323 (2004).
6. Roth, V., Lange, T., Braun, M. & Buhmann, J. A Resampling Approach to Cluster Validation. *COMPSTAT 2002 - Proceedings in Computational Statistics*, 123-128 (2002).
7. Bittner, M. et al. Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature* **406**, 536-540 (2000).
8. Kerr, M.K. & Churchill, G.A. Bootstrapping cluster analysis: Assessing the reliability of conclusions from microarray experiments. *PNAS* **98**, 8961-8965 (2001).
9. Munkres, J. Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics* **5**, 32-38 (1957).
10. Venet, D. MatArray: a Matlab toolbox for microarray data. *Bioinformatics* **19**, 659-660 (2003).
11. Vincent, L. Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. *IEEE Transactions on Image Processing* **2**, 176-201 (1993).
12. Otsu, N. A threshold selection method from grey-level histograms. *IEEE Trans. Systems, Man and Cybernetics* **9**, 62-66 (1979).
13. Vincent, L. & Soille, P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, 583-598 (1991).
14. Jenrich, R.I. Stepwise Discriminant Analysis. *Statistical Methods for Digital Computers*, 76-95 (1977).

15. Duda, R.O., Hart, P.E. & Stork, D.G. Pattern Classification, Edn. 2nd. (John Wiley & Sons, New York; 2001).
16. Dietterich, T.G. Approximate Statistical Test For Comparing Supervised Classification Learning Algorithms. *Neural Computation* **10**, 1895-1923 (1998).
17. Benjamini, Y. & Hochberg, Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)* **57**, 289-300 (1995).