Explainable Multi-Omics Data Fusion for Cancer Subtype Classification and Survival Prediction Using Machine Learning

Data Preprocessing

- Data sourced from linkedomics.org
- Omic, label, and clinical matrices are read from CSV/CSV.GZ
- NaNs are zero-filled
- Only samples common to every modality/label/clinical table are retained so each patient has aligned features across all inputs

Feature Selection

- ANOVA
- RFE
- LASSO
- PCA
- FSD

ANOVA (Analysis of Variances)

ANOVA tests the **null hypothesis**:

H_0 :All group means are equal

against the **alternative**:

H_1:At least one group mean is different.

It does so by comparing:

- **Between-group variability** (how much group means differ from the overall mean)
- **Within-group variability** (how much samples vary inside their own group)

We test whether the mean value of a feature differs across K classes.
Under the null hypothesis H_0:
$\mu_1 = \cdots = \mu_K$, the F−statistic compares variation **between** class means to variation **within** classes.
A large F (small p−value) indicates the feature's class means are not all equal and the feature is discriminative.

Definitions and steps

- **Group means**: $\bar{X}_i$ for class i; **overall mean**: $\bar{X}$
- **Between−group sum of squares (SSB)**: SSB $= \sum_{i=1}^{K} n_i \ (\bar{X}_i - \bar{X})^2$.
- **Within−group sum of squares (SSW)**: SSW = $\sum_{i=1}^{K} \sum_{j=1}^{n_i} (X_{ij} - \bar{X}_i)^2$
- **Degrees of freedom**: $df_1 = K - 1, \ df_2 = N - K$.
- **Mean squares**: MSB $= \frac{SSB}{df_1}$, $MSW = \frac{SSW}{df_2}$.
- **F−statistic**: $F = \frac{MSB}{MSW}$.

p−value (upper−tail)
$p = \Pr(F_{df_1, df_2} \geq F_{obs})$.

Interpretation
Reject H_0 at level α if $p < \alpha$ (equivalently, if $F_{obs} > F_{\alpha;\, df_1, df_2}$); the feature shows significant class separation.

LASSO

Suppose we have data:

$$X \in R^{n \times p}, \quad y \in R^n$$

where n = number of samples, p = number of features.

We want to model:

$$y \approx X\beta + \epsilon$$

where $\beta = [\beta_1, \ldots, \beta_p]^T$ are the regression coefficients.

The **LASSO regression** problem is formulated as:

$$\min_{\beta} \frac{1}{2n} |y - X\beta|_2^2 + \lambda |\beta|_1$$

where:

- $|y - X\beta|_2^2$ is the **residual sum of squares** (model fit term),
- $|\beta|1 = \sum j = 1^p \left|\beta_j\right|$ is the **L1 penalty**,
- $\lambda \geq 0$ is the **regularization parameter** controlling sparsity.

Unlike Ridge regression (L2 penalty), which shrinks coefficients continuously,
LASSO's **L1 penalty** tends to **drive some coefficients exactly to zero**, effectively **performing feature selection**.

So, only a subset of predictors get nonzero weights.

RFE

RFE is a *wrapper method*:

- Fit a model,
- Rank features by importance (weights, coefficients, or importance scores),
- Remove the least important features,
- Refit the model on the remaining features,
- Repeat until a desired number of features is reached.
- This process gives you an ordered ranking of features and can help find the smallest subset that yields good performance.

RFE using Random Forest

Step 1: Train all trees

Each tree in the forest is trained on a bootstrapped sample of the data, with random subsets of features considered at each split.

Step 2: Compute impurity decrease at each node

When a tree splits on a feature (f):

Let:

- (i)= node index
- (I_i)= impurity (e.g., Gini or variance) **before the split**
- $\left(I_{\{i,L\}} \right), \left(I_{\{i,R\}} \right)$ = impurities of the left and right child nodes
- (w_i) = number (or weighted fraction) of samples reaching node (i)
- $\left(w_{i,L}, w_{i,R}\right)$ = same for child nodes

The **impurity decrease** caused by that split is:

$$\Delta_i = w_i I_i - w_{i,L} I_{i,L} - w_{i,R} I_{i,R}$$

- This quantifies how much purer (better separated) the data became after the split.

Step 3: Attribute that decrease to the splitting feature

Each feature accumulates the total impurity decrease across all nodes **where it was used for a split**.

So, for a given feature (f):

$$\text{importance}_f = \sum_{\text{nodes } i \text{ that split on } f} \Delta_i$$

Step 4: Average across all trees

In a Random Forest with (T) trees:

$$\text{importance}_f{}^{(RF)} = \frac{1}{T} \sum_{\text{t=1}}^{\text{T}} \text{importance}_f^{(t)}$$

Step 5: Normalize

Finally, feature importances are normalized so that they sum to 1:

$$\text{importance}_f^{(\text{normalized})} = \frac{\text{importance}_f^{(RF)}}{\sum_j \text{importance}_j^{(RF)}}$$

PCA (Principal Component Analysis)

Given a dataset $X \in R^{n \times p}$ (with n samples and p features), PCA seeks **orthogonal directions (principal components)** that capture the **maximum variance** in the data.

Each principal component is a **linear combination** of the original features:

$$z_k = X\, w_k$$

where w_k is a weight vector (eigenvector).

The goal is:

$$\max_{w_k} \mathrm{Var}(z_k) = w_k^{\top} S w_k$$

subject to $|w_k| = 1$ and $w_k^{\top} w_j = 0$ for $j < k$,

where $S = \frac{1}{n-1} X^{\top} X$ is the covariance matrix.

Solving:

$$S w_k = \lambda_k w_k$$

gives:

- w_k: direction of the k^{th} principal component (eigenvector)
- λ_k : variance explained by that component (eigenvalue)

Thus, the **total variance** of the dataset is decomposed into parts corresponding to each component.

Feature Selection using PCA

Each principal component $w_k = [w_{1k}, w_{2k}, \ldots, w_{pk}]^{\top}$ contains the **loading values** i.e., how strongly each original feature contributes to that component.

We can:

1. Choose the top K components that explain most variance.
2. For each feature j, compute its **total contribution**:
3. $s_j = \sum_{k=1}^{K} |w_{jk}|$
4. Select features with the largest s_j.

These are the features **most influential** in the directions of maximum variance.

Feature Selection with Distribution

The **KS test** is a non-parametric test that compares two distributions. It returns a **p-value**, which helps test if two distributions are **statistically similar**.

Test 1: $\boldsymbol{p_1 \leftarrow KS(X_{sub}, X)}$

- Compares the subset distribution to the entire dataset.
- If $p_1 > k$, the subset resembles the global distribution (i.e., no significant difference).
- Ensures the feature is stable in random subsets.

Test 2: $\boldsymbol{p_2 \leftarrow KS(X_a, X_b, \dots, X_n)}$

- Compares distributions across **clinical subgroups** (e.g., cancer types).
- Low p_2 means **feature varies across classes** — indicating potential **discriminative power**.
- But for a **low-noise feature**, we want **less variation** across classes → So we keep it **only if** $\boldsymbol{p_2 < k}$.

Test 3: $\boldsymbol{p_3 \leftarrow KS(X_{sub,a}, X_{sub,b}, \dots, X_{sub,n})}$

- Similar to p_2, but applied only to the subset.
- Ensures that the subset maintains clinical consistency.

A feature was considered to be low-noise and high-informative if $p_1 > k, p_2 < k, p_3 < k$

Kolmogorov-Smirnov Test

Instead of comparing means or variances, K–S compares **entire distributions** by quantifying how far apart their cumulative curves are.

For two CDFs:

$$G_m(x) = \text{empirical CDF of sample 2 (or theoretical CDF if one-sample)} F_n(x) = \text{empirical CDF of sample 1}$$

The **K–S statistic** is:

$$D_{n,m} = \sup_x |F_n(x) - G_m(x)|$$

That is, the **maximum vertical distance** between the two CDFs.

Empirical CDF Definition

For a sample $X_1, X_2, \ldots, X_n$, the **empirical cumulative distribution function (ECDF)** is:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \le x)$$

where $I(\cdot)$ is an indicator function (1 if true, else 0).

The **p-value** is computed as:

$$p = Q_{KS}(\sqrt{n} D_n)$$

where Q_{KS} is the survival function (1 - CDF) of the Kolmogorov distribution.

Models Used

- SVM
- Random Forest
- XGBoost
- MLP
- Attention MOI

Random Forest (RF)

RF is an ensemble of decision trees trained on bootstrapped samples with feature subsampling at each split; predictions are the average of tree probabilities. Splits commonly optimize **Gini impurity**, $\mathrm{Gini} = 1 - \sum_c p(c)2$, or entropy. .

Split criterion & prediction

$\mathrm{Gini} = 1 - \sum_c p(c)2$; RF prediction is average of tree probabilities.

In our pipeline we feed RF compact, noise-reduced panels produced by FSD + feature selection. RF captures nonlinear interactions across omics with minimal tuning and provides straightforward importance scores to highlight influential biomarkers.

Support Vector Machine

SVM finds the decision boundary with the largest margin between classes.
Works well with high-dimensional omics and small N.

Soft-margin primal

$$\begin{aligned} \min_{w,b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n, \\ & \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

We use linear or RBF kernels after curating features with FSD and filters (ANOVA/RFE/LASSO/PCA). Features are standardized and probability estimates are obtained from the fitted model to compute AUC. SVM is a strong choice for high-dimensional, low-sample omics and serves as a robust baseline for multi-omics classification and risk stratification.

XGBoost (gradient boosting trees)

XGBoost builds an *additive* model of shallow trees: $\hat{y}^T = \sum_{\{t=1\}}^{T} f_{t(x)}$, learned by minimizing a regularized objective
$\sum_i l(y_i, \hat{y}_i) + \sum_t \Omega(f_t)$, with second-order gradients, shrinkage, and column subsampling.

Additive objective

$$minimize\ \Sigma\ l(y, \hat{y}) + \ \Omega(f_t)\ wrt\ \ \hat{y}^T = \ \Sigma\ f_{t(x)}\ .$$

We train it on the same FSD-filtered feature panels, typically achieving high AUC with relatively few features. Its regularization resists overfitting on small cohorts.

Deep Neural Network (MLP)

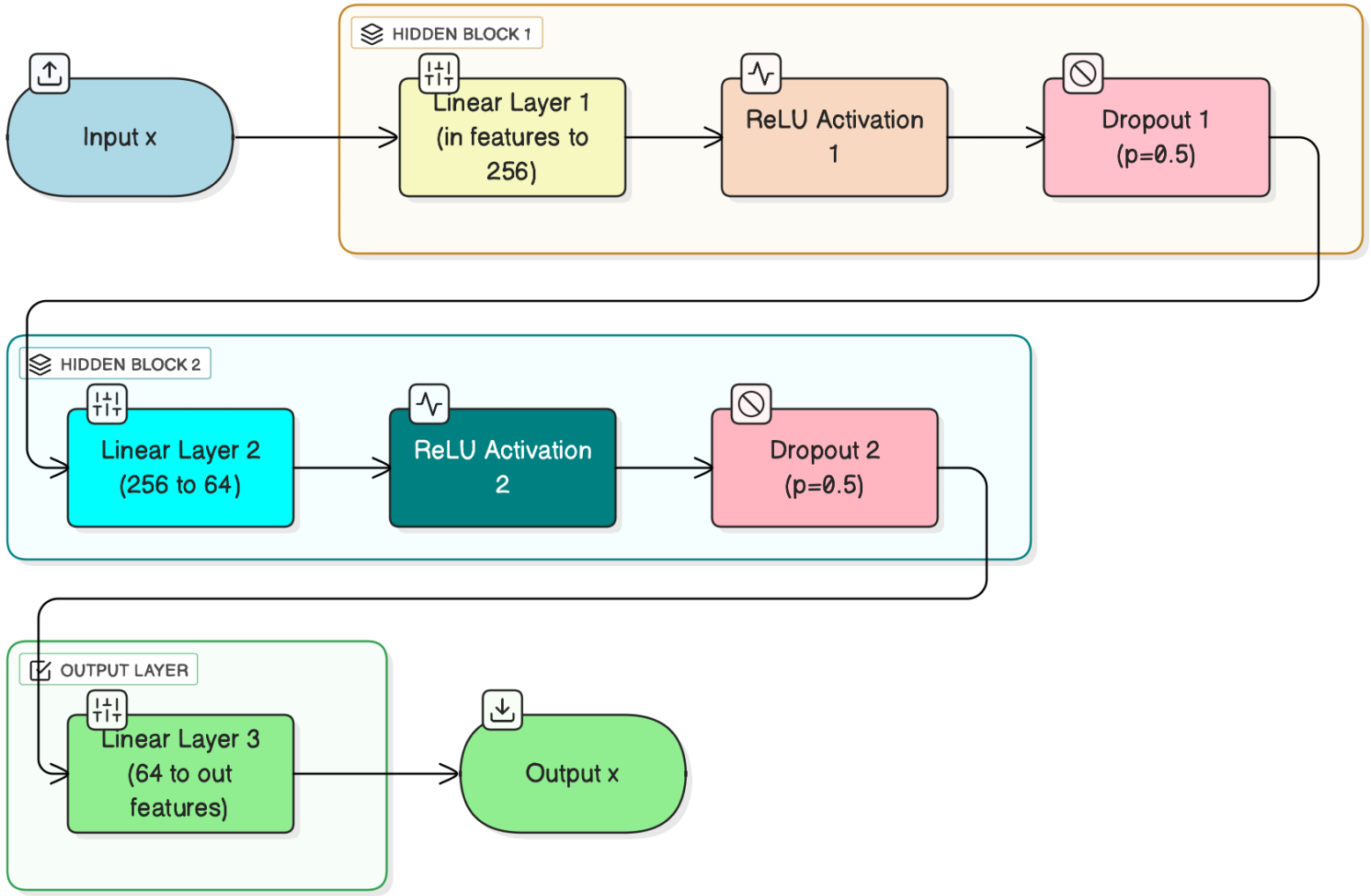
Our deep classifier is a feed-forward network applied to the concatenated multi-omics vector trained with cross-entropy $L = -\sum y \cdot \log \hat{y}$, dropout, and Adam optimizer.

Forward + loss

$$h_1 = ReLU(XW_1 + b_1), \ldots, \hat{y} = softmax(h_k W_k + b_k);\ L = -\Sigma\, y \cdot \log(\hat{y})\,.$$

We optionally prefilter inputs with FSD and PCA/ANOVA/RFE/LASSO to reduce noise and dimensionality. The MLP models nonlinear cross-omic interactions; we log per-epoch metrics (ACC/AUC/F1/Recall/Precision)

Multi Layer Perceptron (MLP)



Attention Multi-Omics Integration (MOI)

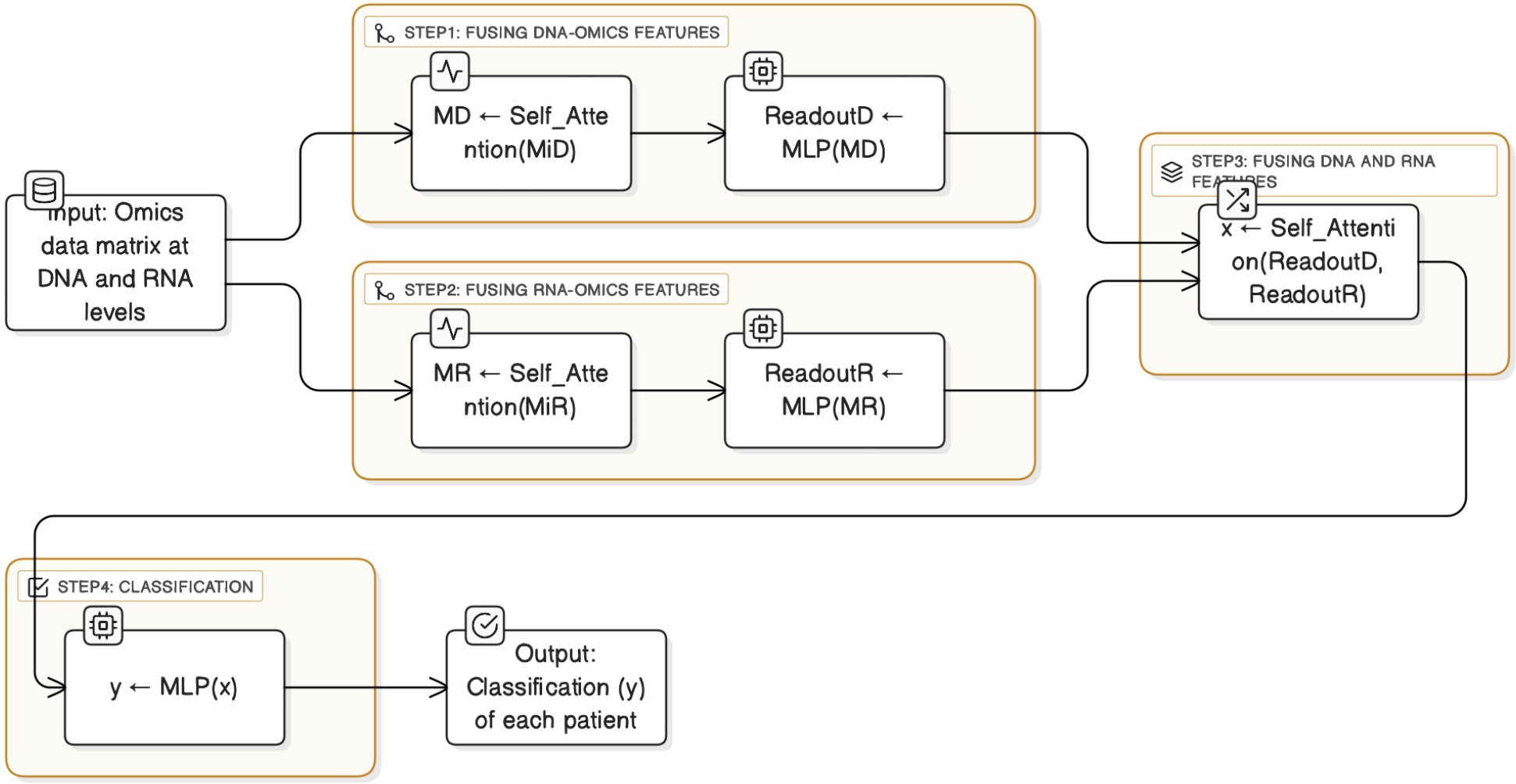
AttentionMOI learns separate embeddings for DNA-level signals (e.g., methylation + CNV) and RNA expression, then fuses them for classification. Its core mechanism is scaled dot-product attention which lets the model weight informative features across omics before a final MLP + softmax layer.

Scaled dot-product attention

$$\mathrm{Attn}(\mathrm{Q},\mathrm{K},\mathrm{V}) = \mathrm{softmax}\left(\frac{\mathrm{QK}^{\mathrm{T}}}{\sqrt{\mathrm{d}}}\right)\mathrm{V}$$

fuse DNA/RNA → MLP → softmax

Attention Multi-Omics Integration (MOI)



Metrics used

Assume we have K *classes* $\{1, 2, \dots, K\}$, and a confusion matrix $M \in R^{K \times K}$, where
$M_{ij} =$ number of samples whose **true class** is i and **predicted class** is j.

1. Accuracy

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} = \frac{\sum_{k=1}^{K} M_{kk}}{\sum_{i=1}^{K}\sum_{j=1}^{K} M_{ij}}$$

2. Precision (per class and averaged)

For class k:

$$\text{Precision}_k = \frac{M_{kk}}{\sum_{i=1}^{K} M_{ik}}$$

Macro Precision (unweighted average):

$$\text{Precision}_{\text{macro}} = \frac{1}{K}\sum_{\text{k=1}}^{\text{K}} \text{Precision}_k$$

3. Recall (Sensitivity, per class and averaged)

For class k:

$$\text{Recall}_k = \frac{M_{kk}}{\sum_{j=1}^{K} M_{kj}}$$

Macro Recall: $\text{Recall}_{\text{macro}} = \frac{1}{K}\sum_{\text{k=1}}^{\text{K}} \text{Recall}_k$

4. F1 Score

For class k:

$$\text{F1}_k = \frac{2 \times \text{Precision}_k \times \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k}$$

Macro F1 Score: $\text{F1}_{\text{macro}} = \frac{1}{K}\sum_{\text{k=1}}^{\text{K}} \text{F1}_k.$

5. AUC (Area Under ROC Curve) for Multi-Class

In the **one-vs-rest (OvR)** scheme:

For each class k, compute:

- **True Positive Rate (TPR):**

$$\text{TPR}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k}$$

- **False Positive Rate (FPR):**

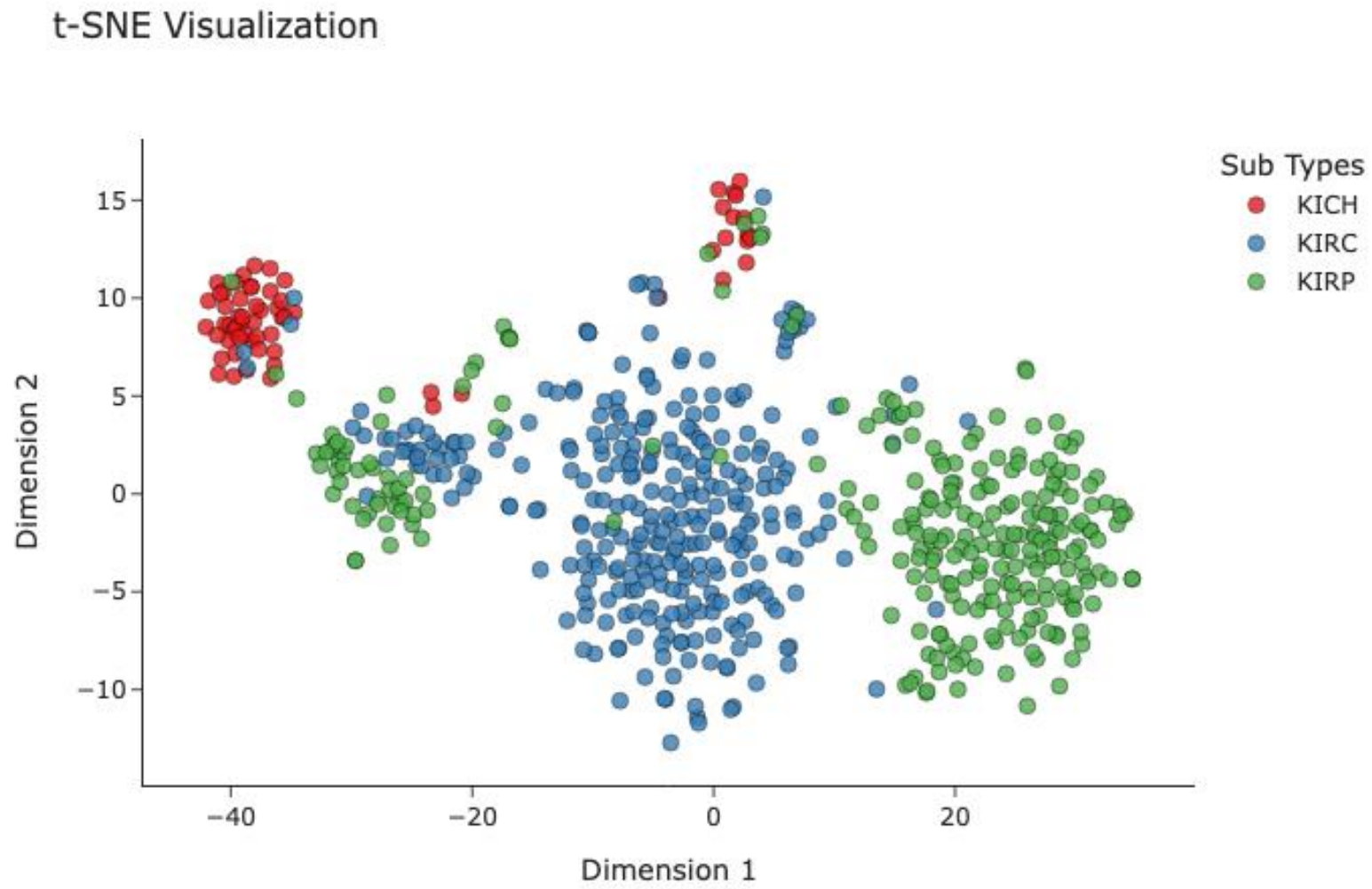
$$\text{FPR}_k = \frac{\text{FP}_k}{\text{FP}_k + \text{TN}_k}$$

Then compute the **AUC for each class**
$AUC_k = \int_0^1 TPR_k(FPR_k), d(FPR_k).$

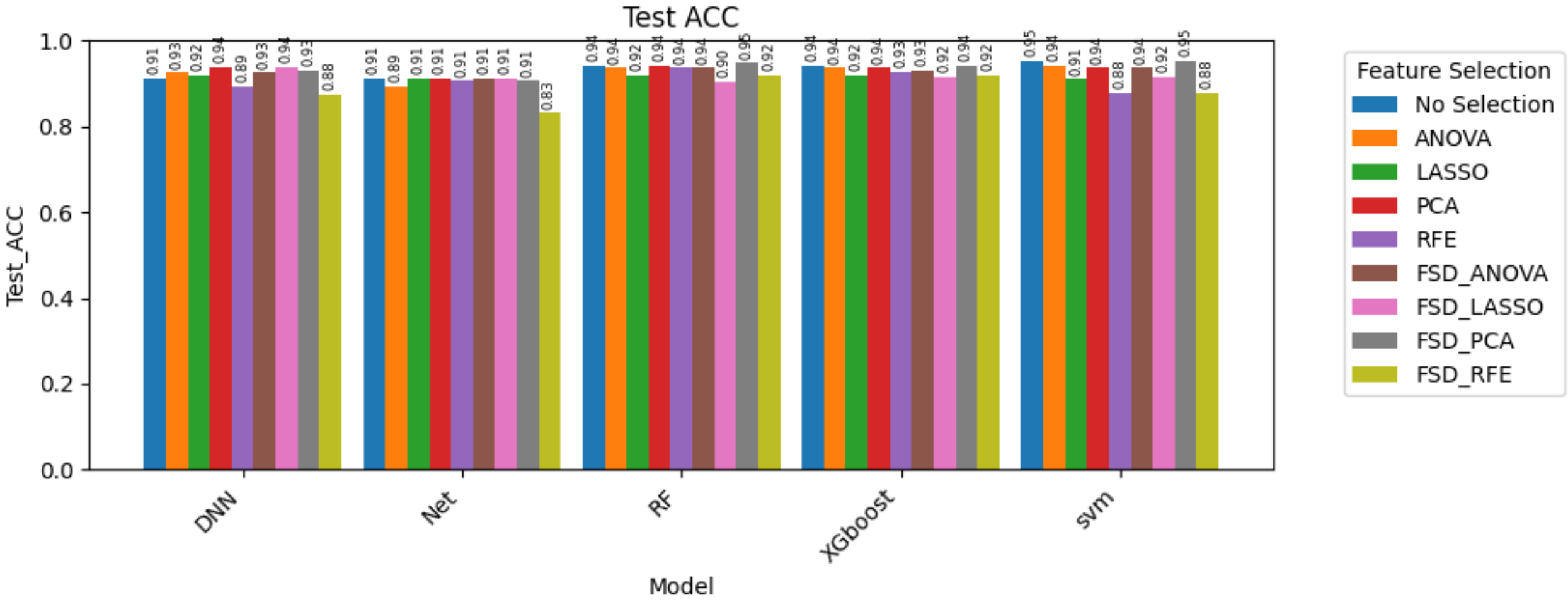
Finally:

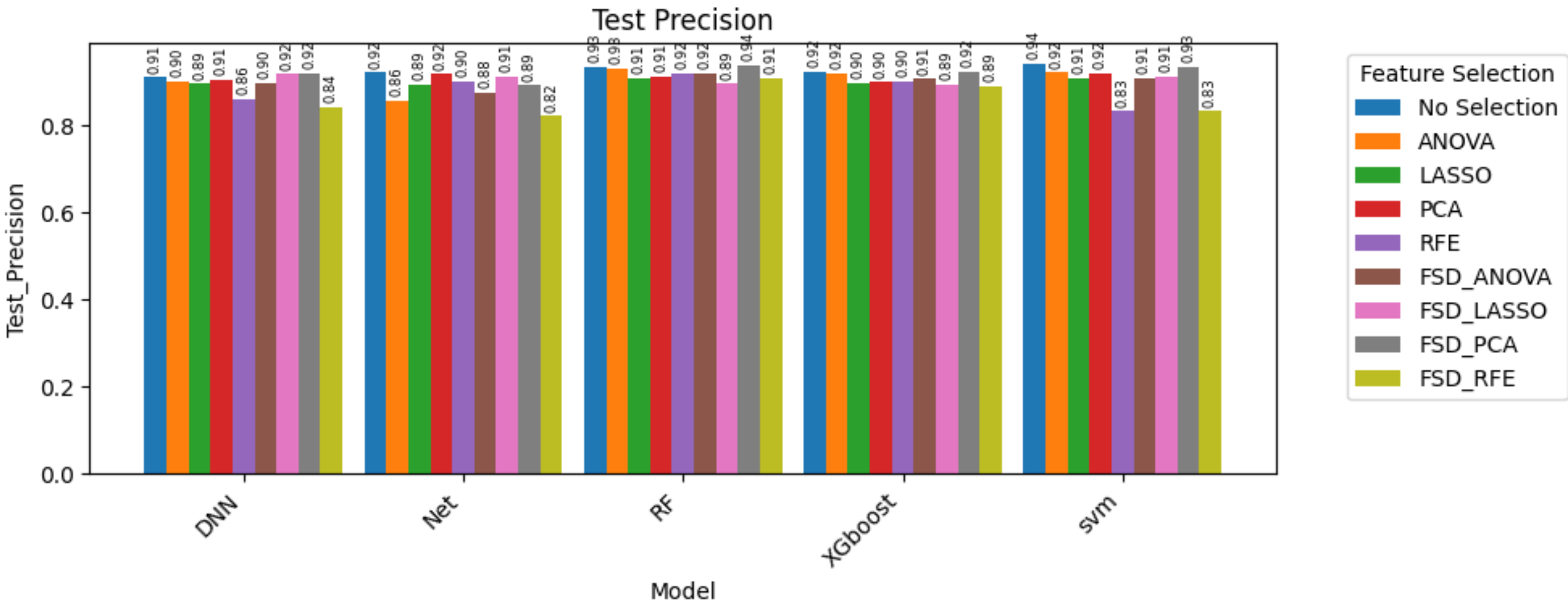
$$\text{AUC}_{\text{macro}} = \frac{1}{K}\sum_{\text{k=1}}^{\text{K}} AUC_k$$

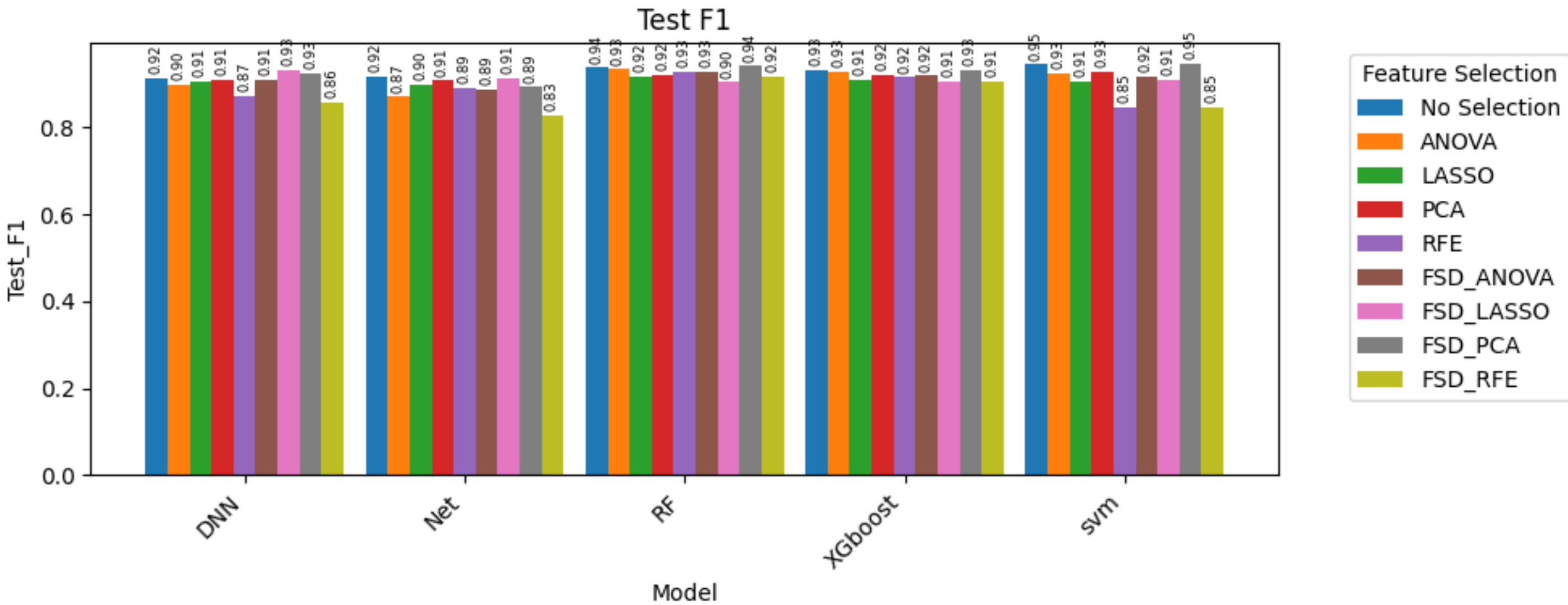
Pan-kidney cohort (TCGA-KIPAN)

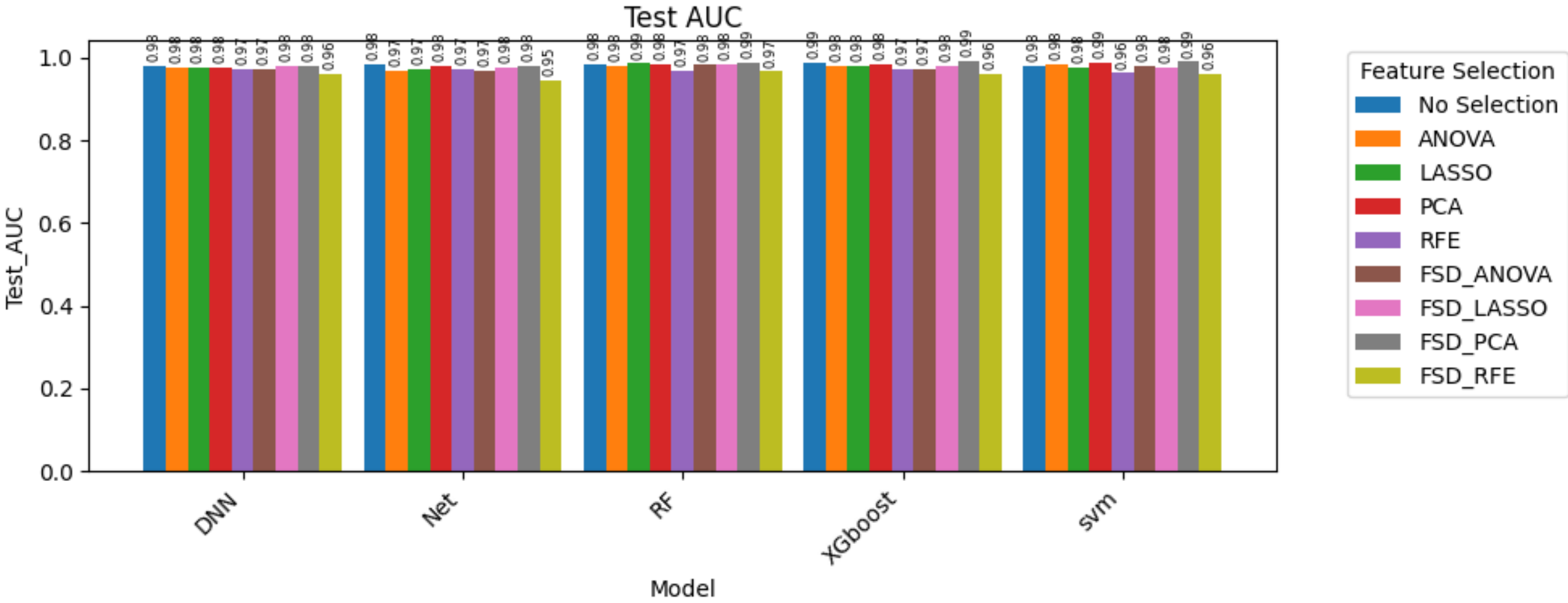


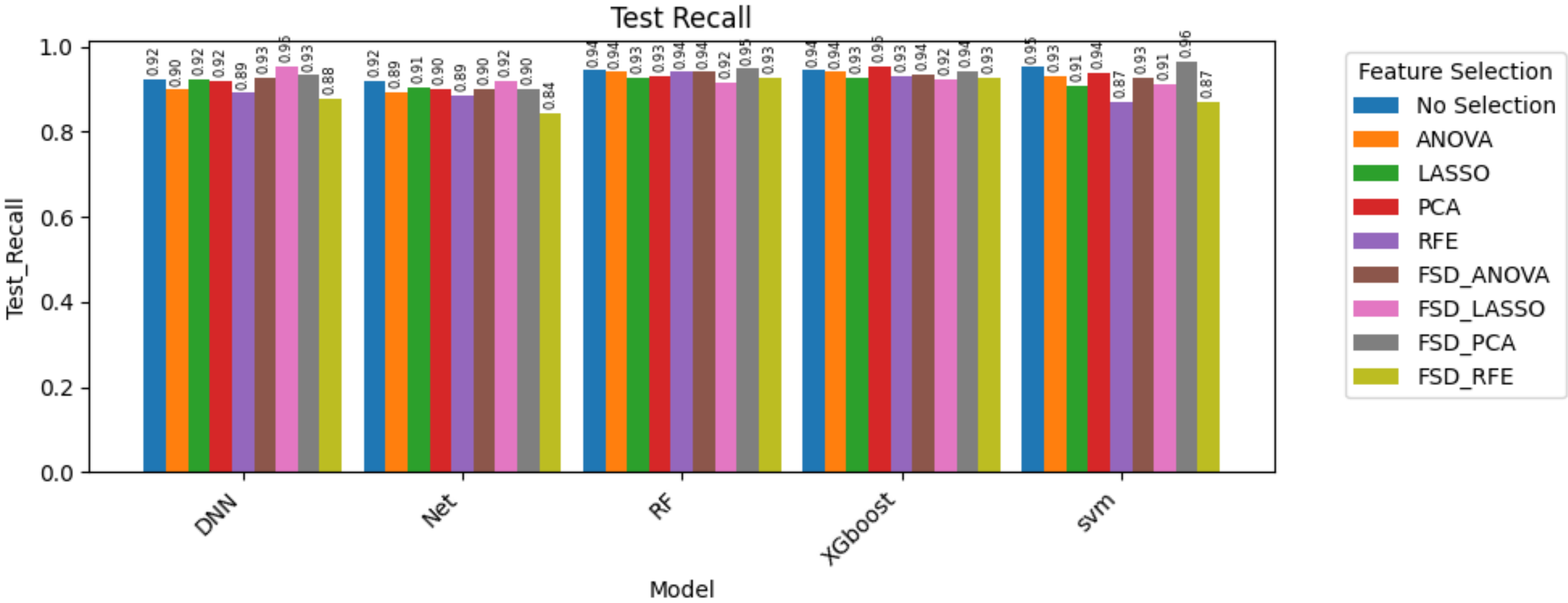
Results for Subtype Classification (KIPAN)



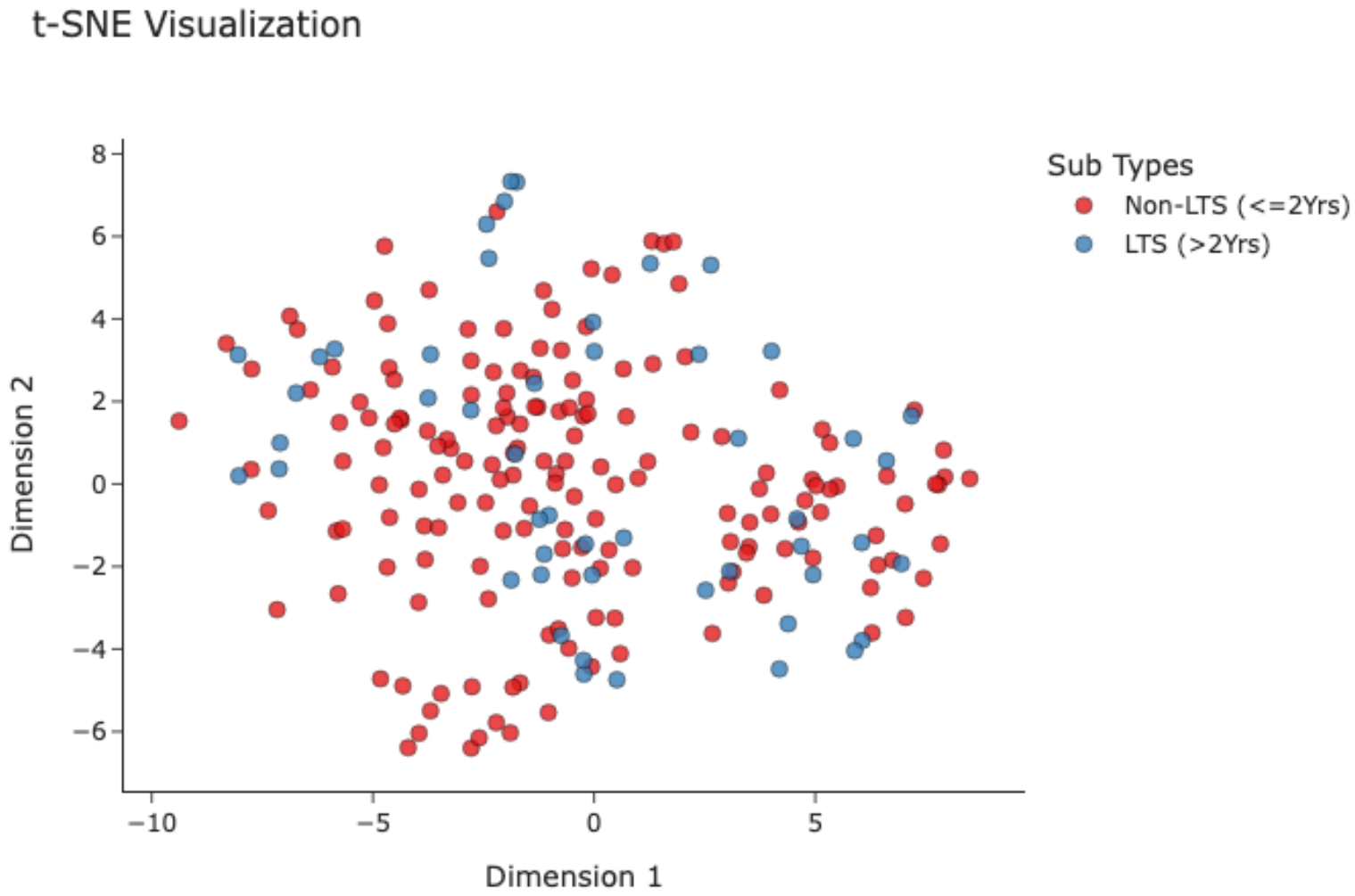

Test Precision
Test_Precision
0.0
0.2
0.4
0.6
0.8
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
ANOVA
LASSO
PCA
RFE
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE


Test F1
Test_F1
0.0
0.2
0.4
0.6
0.8
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
ANOVA
LASSO
PCA
RFE
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE

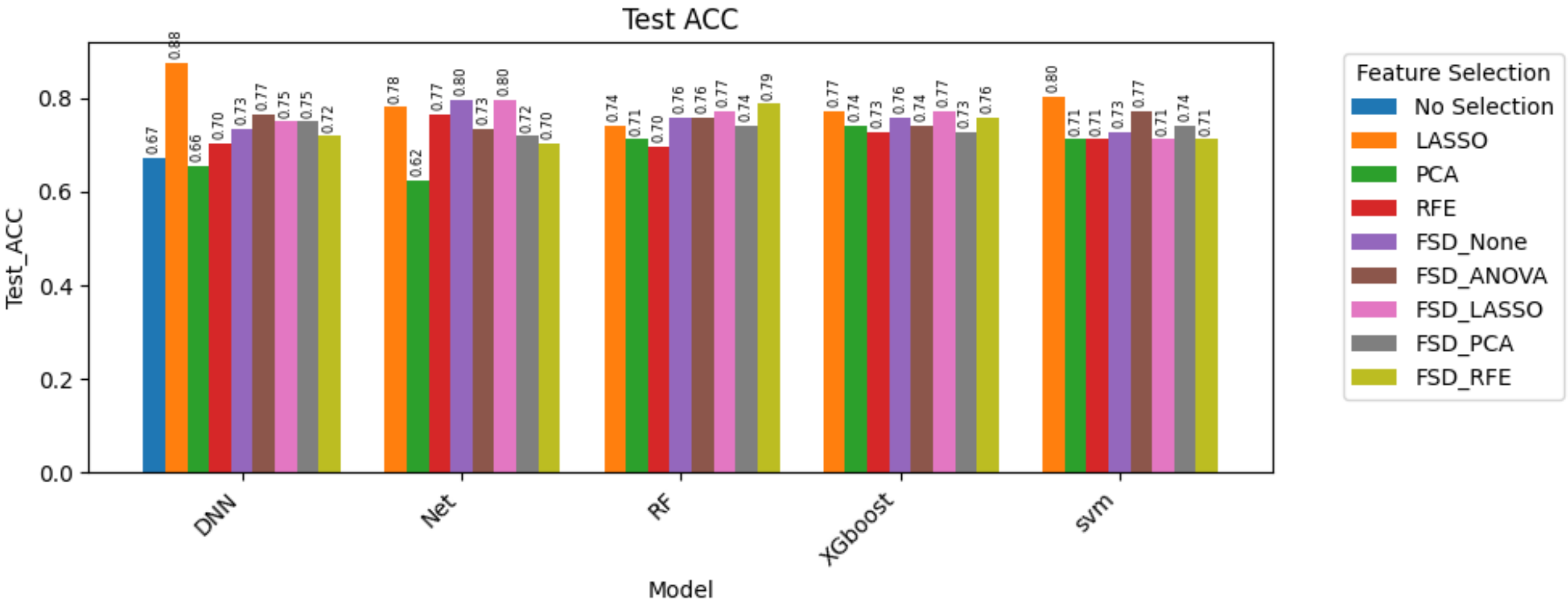

Test AUC
Test_AUC
1.0
0.8
0.6
0.4
0.2
0.0
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
ANOVA
LASSO
PCA
RFE
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE

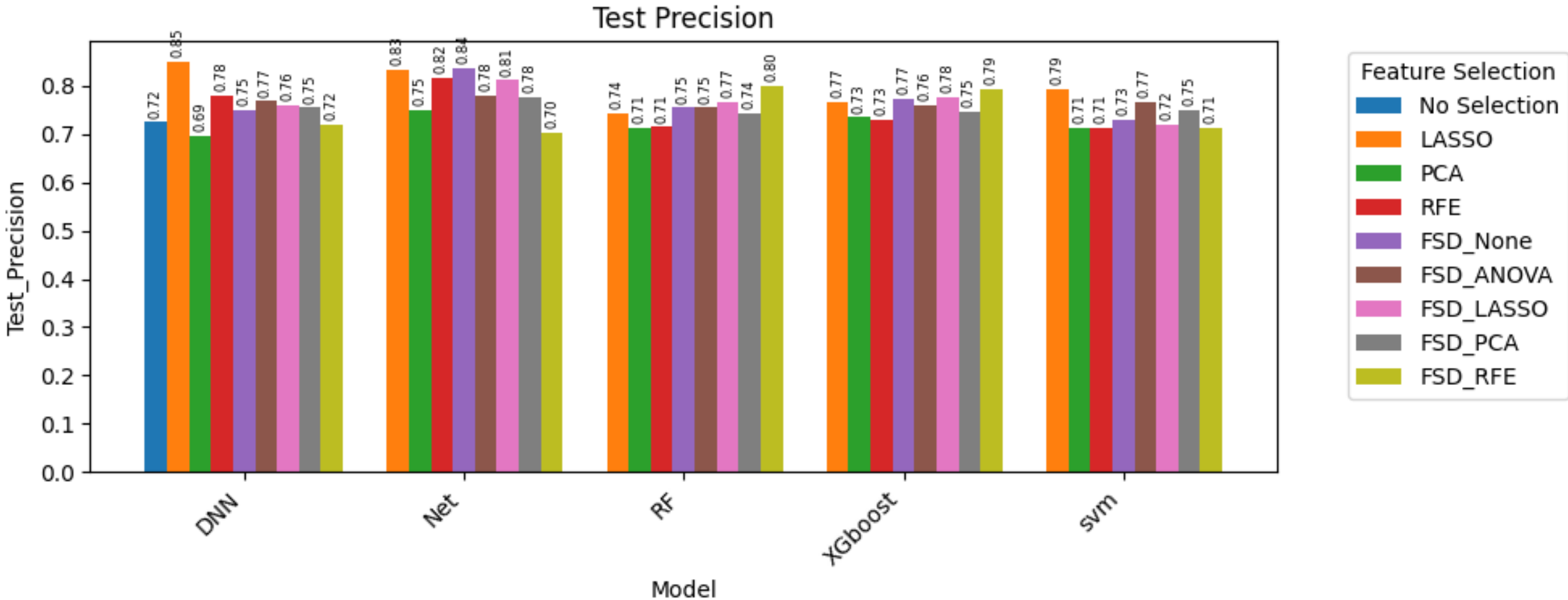

Test Recall
Test_Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.92 0.90 0.92 0.92 0.89 0.93 0.95 0.93 0.88
0.92 0.89 0.91 0.90 0.89 0.90 0.92 0.90 0.84
0.94 0.94 0.93 0.93 0.94 0.94 0.92 0.95 0.93
0.94 0.94 0.93 0.95 0.93 0.94 0.92 0.94 0.93
0.95 0.93 0.91 0.94 0.87 0.93 0.91 0.96 0.87
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
ANOVA
LASSO
PCA
RFE
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE

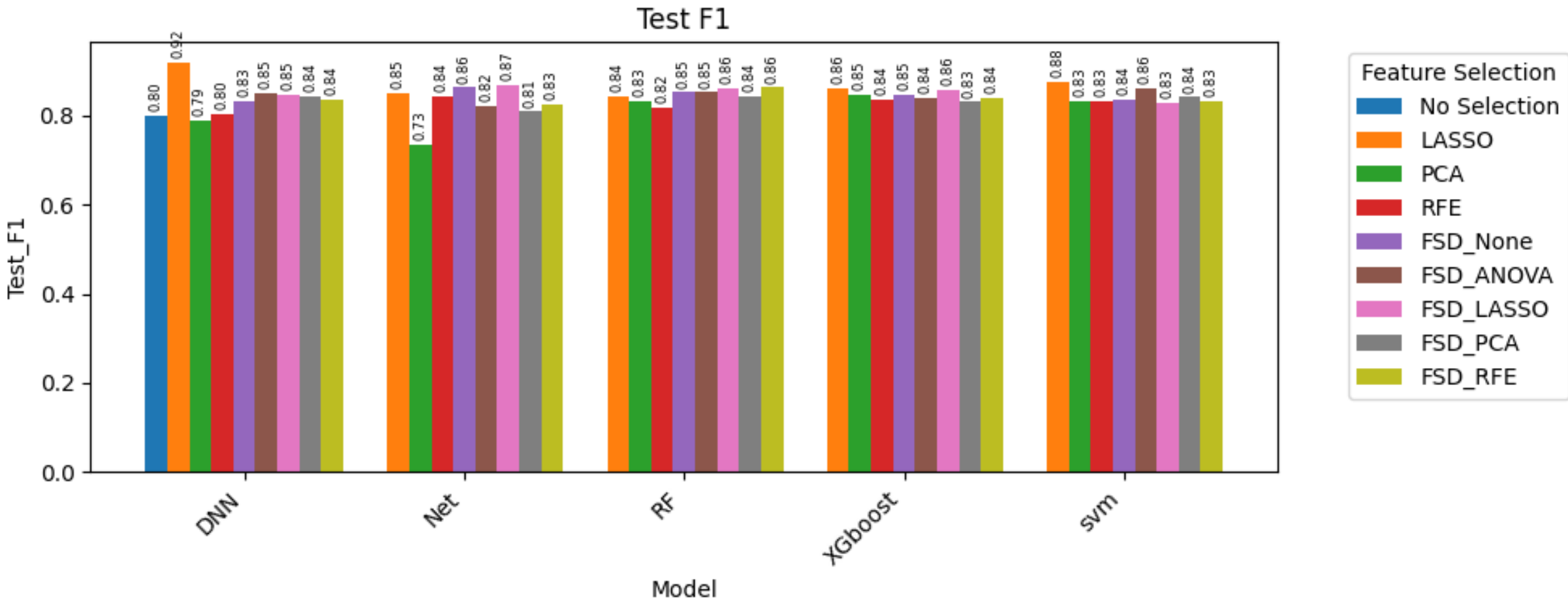
Glioblastoma multiforme (TCGA-GBM)

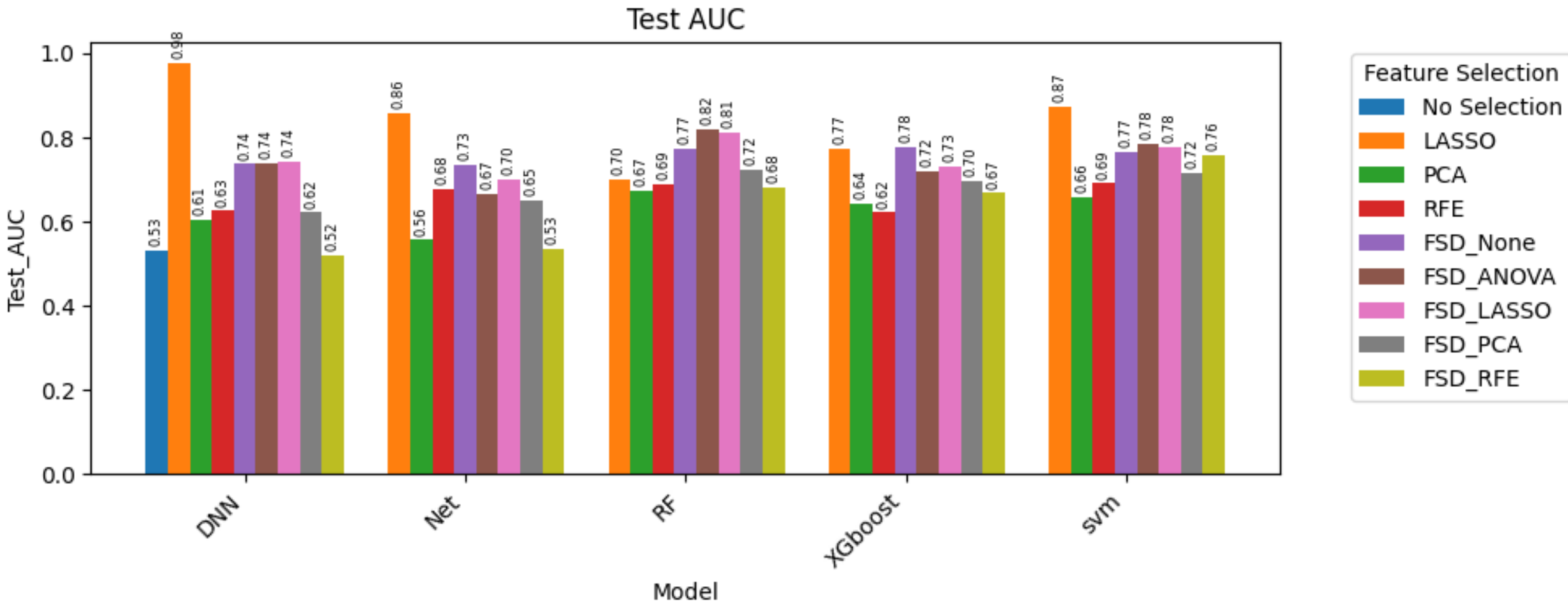


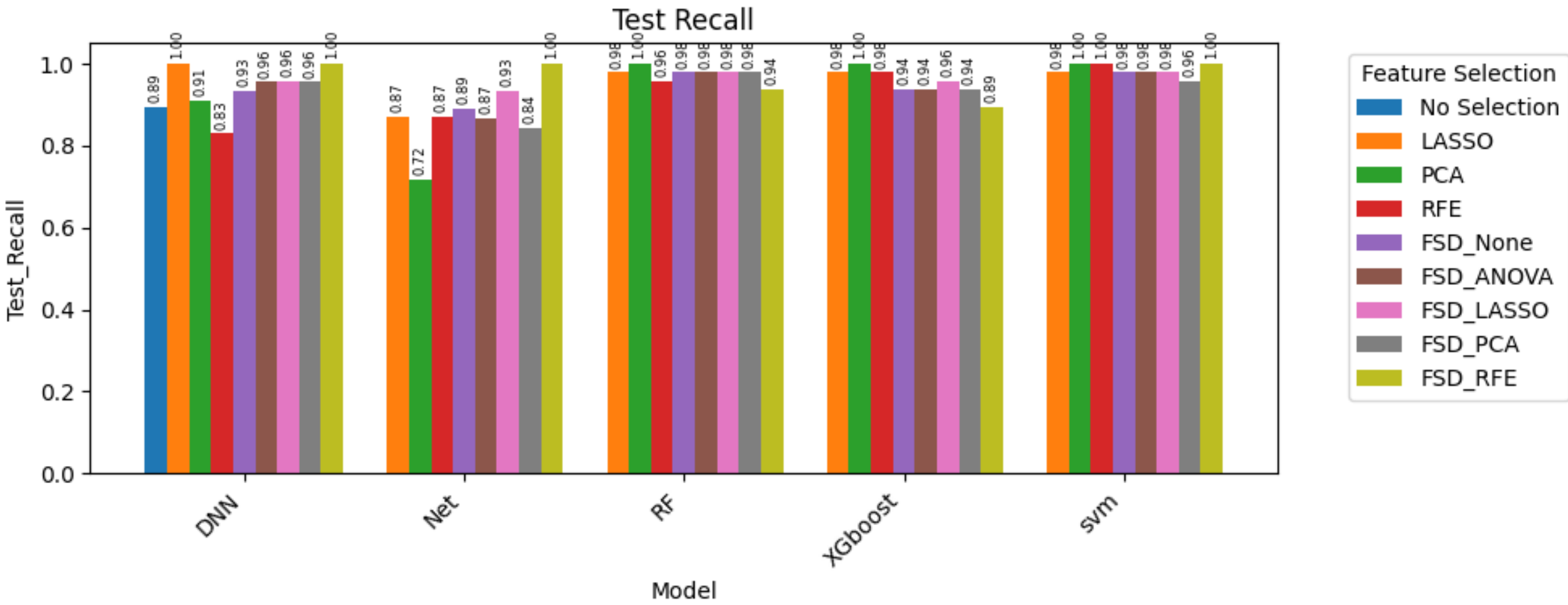
Results for Survival Prediction (GBM)




Test Precision
Test_Precision
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.72
0.85
0.69
0.78
0.75
0.77
0.76
0.75
0.72
0.83
0.75
0.82
0.84
0.78
0.81
0.78
0.70
0.74
0.71
0.71
0.75
0.75
0.77
0.74
0.80
0.77
0.73
0.73
0.77
0.76
0.78
0.75
0.79
0.79
0.71
0.71
0.73
0.77
0.72
0.75
0.71
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
LASSO
PCA
RFE
FSD_None
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE

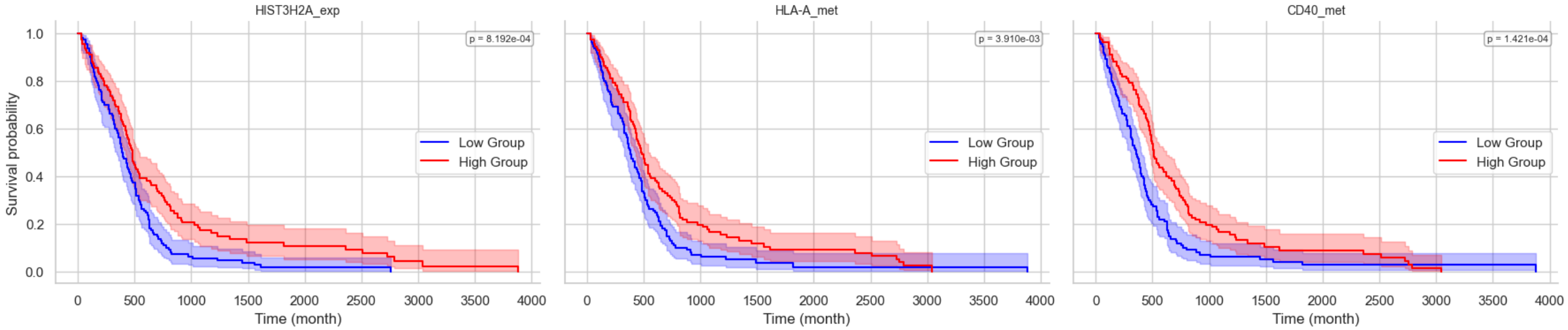

Test F1
Test_F1
0.0
0.2
0.4
0.6
0.8
0.80
0.92
0.79
0.80
0.83
0.85
0.85
0.84
0.84
0.85
0.73
0.84
0.86
0.82
0.87
0.81
0.83
0.84
0.83
0.82
0.85
0.85
0.86
0.84
0.86
0.86
0.85
0.84
0.85
0.84
0.86
0.83
0.84
0.88
0.83
0.83
0.84
0.86
0.83
0.84
0.83
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
LASSO
PCA
RFE
FSD_None
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE


Test AUC
Test_AUC
1.0
0.8
0.6
0.4
0.2
0.0
0.53
0.98
0.61
0.63
0.74
0.74
0.74
0.62
0.52
0.86
0.56
0.68
0.73
0.67
0.70
0.65
0.53
0.70
0.67
0.69
0.77
0.82
0.81
0.72
0.68
0.77
0.64
0.62
0.78
0.72
0.73
0.70
0.67
0.87
0.66
0.69
0.77
0.78
0.78
0.72
0.76
DNN
Net
RF
XGboost
svm
Model
Feature Selection
No Selection
LASSO
PCA
RFE
FSD_None
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE


Test Recall
Test_Recall
Model
DNN
Net
RF
XGboost
svm
Feature Selection
No Selection
LASSO
PCA
RFE
FSD_None
FSD_ANOVA
FSD_LASSO
FSD_PCA
FSD_RFE

Kaplan Meier Plot

Keplan-Meier curves of top three features selected by RFE + FSD in GBM data. *P*-value calculated through Log-Rank test



Indian Landscape

ICGA (Indian Cancer Genome Atlas) is the flagship, nationwide effort building India-specific, multi-omics cohorts and a cBioPortal-based data portal. The first release focuses on **50 Indian breast-cancer patients** with WGS (~70×), RNA-seq and proteomics generated; the public portal currently exposes **somatic mutation** layers with controlled access via a DAC process.

Indian Cohort Datasets

1. **ICGA Breast Cancer Cohort (India) — multi-omics (DNA/RNA/proteins)**
   Full multi-omics/clinical available on request (DAC). Built on cBioPortal.

2. **dbGENVOC (Indian Oral Cancer) — large variant resource**

Open, queryable database with ~24M somatic/germline variants from Indian OSCC (exomes n≈100; genomes n≈5). Raw sequences reside in EGA (EGAS accessions listed in paper).