# Mutation Profile to Predict Tumor Stage in Lung Adenocarcinoma

1st Calvin Kuo
*Mechanical Engineering Department*
*Stanford University*
Stanford, USA
calvink@stanford.edu

***Abstract*—Lung adenocarcinoma is among the most common and deadliest cancers in the United States, accounting for an estimated 120,000 new cases every year, and responsible for nearly 10% of cancer deaths [7]. Because of its prevalence and high mortality rate, it is important to understand the histological progression of lung cancers. In this work, I classified tumor stage based on mutations in genes with substantial mutation frequency in lung adenocarcinoma. I trained a support vector machine classifier and a 3-layer neural network on a 501 sample training dataset to achieve classification accuracies of 60% on an unbiased 159 sample test set. However, the classifiers were prone to overfit on the training set, which typically yielded training accuracies in excess of 80%, despite utilizing feature space reduction methods such as principal component analysis. I also developed a 4-layer neural network architecture which better mapped mutations to the genes they affect, which achieved 69.4% test set accuracy. My models also predicted that genes such as ZMYND10, which control cell proliferation, are important predictors of tumor stage, and may further indicate a tumor's aggressiveness. Findings here can be used to evaluate the potential aggressiveness of early stage tumors, and inform treatment regimens to prevent the spread of cancer and improve prognosis.**

## I. Introduction and Related Work

Many cancers are the result of genetic mutations that cause uncontrolled proliferation of otherwise healthy cells. Lung cancers have been traditionally linked to environmental conditions such as smoking [4], which cause mutations throughout the genome [10]. With sequencing technology becoming faster, cheaper, and more accurate, health agencies have organized databases of sequenced cancer tumors to help researchers identify underlying gene mutations patterns in various cancers. Previous studies that sequenced the whole genome of lung adenocarcinoma tumors have identified several genes that are commonly mutated [3], [9]. Others have utilized unsupervised clustering techniques on protein expression to further differentiate tumors [1], [2], and simple linear regressions to relate total number of mutations to tumor progression [3].

While researchers have devoted resources to identifying the presence of cancer from genome sequencing, little effort has been expended to predict the progression of existing tumors. Such information is important to patients who already have cancer, as a detailed understanding of the aggressiveness of their tumor may inform better treatment options and improve prognosis. As an example, patients with early stage, but aggressive tumors may receive more advanced treatments to prevent substantial spread and improve prognosis.

Unfortunately, there is no diagnostic classification for the aggressiveness of a lung adenocarcinoma tumor, primarily because there is little longitudinal data tracking the progression of tumors to determine their aggression. Instead, lung adenocarcinoma tumors are typically classified with a tumor stage based on their extent throughout the lung and body at the time of diagnosis. For this project, I used this tumor stage classification as a representation of tumor aggressiveness, with later stage tumors that have spread significantly representing aggressive tumors. *Thus, I used the mutations within commonly mutated genes as input into a support vector machine and neural networks to output tumor stage.*

## II. Dataset and Features

### A. Dataset

A publicly available dataset containing data from 660 lung adenocarcinoma samples in two different studies was used for this work (cBioPortal: www.cbioportal.org) [3], [6], [9]. I used the larger dataset containing 501 samples for training and validation purposes, and the smaller dataset containing 159 samples for testing purposes. In both datasets, tumor stages were classified by trained physicians at the time of diagnosis, and tumor mutations were identified by comparing the tumor genome sequence against the patient wild type sequence taken from another part of the body.

### B. Binary Tumor Stage Labels

Lung adenocarcinoma tumor stages are typically defined based on the American Joint Committee on Cancer (AJCC) TNM system [5], which describes a tumor based on its size (T), whether it has spread to adjacent lymph nodes (N), and whether it has metastasized (M). With this tumor description, the tumor is then binned into one of seven stages: IA, IB, IIA, IIB, IIIA, IIIB, and IV, which can be further broken down into four primary stages describing the tumor's location (Figure 1).

To further simply the classification problem, I chose to group the tumor stages for binary classification, with one label comprising tumors in stages IA and IB (localized to one lung), and the other label comprising tumor stages IIA, IIB, IIIA, IIIB, and IV (spread or metastasized). Conveniently, this binary division also gave a roughly 50/50 distribution between

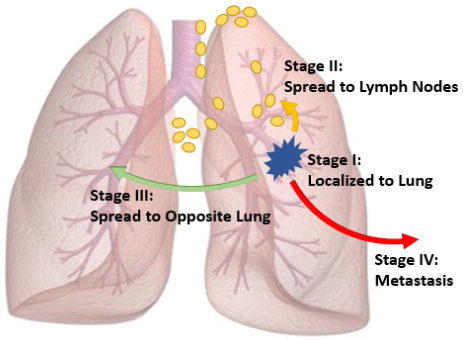


Fig. 1. Tumor staging based on tumor location within the body. (I) Tumor is localized to one lung. (II) Tumor has spread to adjacent lymph nodes. (III) Tumor has spread to the opposite lung. (IV) Tumor has metastasized to other parts of the body. For binary classification, tumors in stage I were given one label, and tumors in stages II, III, and IV were given the other label.

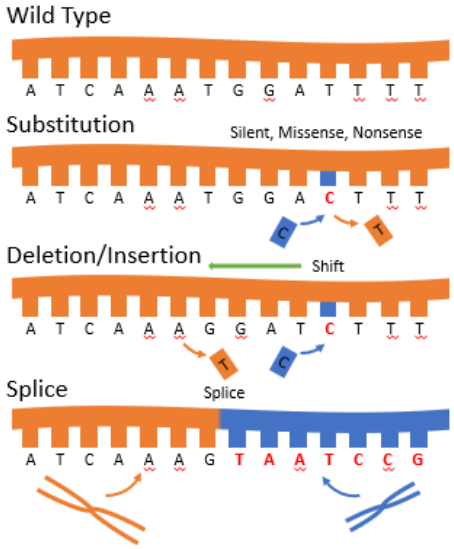


Fig. 2. Different mutation types have varying effects on downstream protein coding. Base pair substitutions can have no effect on protein coding (silent), change one amino acid (missense), or code a stop codon that halts protein production (nonsense). Base pair insertion and deletion can cause shifts in the genome sequence, having drastic effects on downstream amino acid coding. Finally, splice mutations occur when genome sequences from different chromosomes are stitched together, creating fusion genes.

the two labels in both the training and test datasets (265 IA or IB out of 501 samples in training set, and 71 IA or IB out of 159 samples in test set).

### C. Feature Set

In the 501 training set samples, there were over 150,000 mutations identified in over 17,000 genes. Because cancers are typically associated with mutations in specific genes that cause unusual proliferation, I focused on mutations that were present in genes that were significantly mutated in lung adenocarcinomas. Previous literature has identified 36 genes that are significantly mutated in lung adenocarcinomas [3], [9], and I added an additional 32 genes that were mutated in over 25% of the 501 sample training dataset. This reduced the number of genes to 68, and the number of mutations to 11,943.

For each mutation, I extracted information about the gene in which the mutation was located (of the 68 significantly mutated genes described previously), and the effect the mutation had on the resulting amino acid codon. Briefly, there are several types of mutations that can alter the coding of amino acids and production of proteins (Figure 2). Simple nucleotide substitutions alter single base pairs, and can either change the resulting amino acid at the mutation location (missense), stop protein production by coding the stop codon (nonsense), or have no effect (silent). More complicated insertion or deletion of base pairs shift entire coding sequences, and even more complex mutations known as splice mutations occur when two distinct sequences from separate chromosomes are stitched together, resulting in fusion genes.

From this mutation information, I generated three feature sets. The first feature set described the number of mutations present within each of the 68 genes. The second feature set described the number of each type of mutation effect (missense, nonsense, shift, and splice) occurring within all 68 genes. Finally, I included a third feature set, which described the number of mutation effects separately within each of the 68 genes, yielding the largest feature set with $4 * 68 = 272$ features. I created these feature sets, because I hypothesized that they would best describe which protein structures were altered (through gene mutation), and how significantly they may have been altered (through mutation effect). I did not include silent mutations because they do not alter protein coding. In addition, I standardized features using the mean and standard deviation of training dataset features before further processing and training.

### D. Feature Space Reduction

While I was able to organize mutations into meaningful feature sets, the size of the feature sets was large compared to the relatively small datasets. Thus, I needed to employ feature space reduction techniques to prevent possible over-fitting.

For the first technique, I trained a Naive-Bayes multinomial event model with Laplace smoothing on the training dataset, and extracted the 25 most predictive features from each feature set. Briefly, Naive-Bayes relies on the assumption that the presence of each mutation feature given the tumor stage label is conditionally independent of the presence of other mutation features given the tumor stage label. Maximizing $\ell(\phi)$ the log of the joint likelihood over the samples in the training dataset (dataset size $m$) parameterized by estimates of the conditional probabilities $\phi_{i|y=1} = p(x_i = 1|y = 1)$, $\phi_{i|y=0} = p(x_i = 1|y = 0)$ and label distribution $\phi_y = p(y = 1)$:

$$\ell(\phi) = \sum_{i=1}^{m}\sum_{j=1}^{n} \log p(x_j^{(i)}|y; \phi_{i|y=0}, \phi_{i|y=1}) + \sum_{i=1}^{m} \log p(y^{(i)}; \phi_y) \quad (1)$$

I can solve for the conditional probability estimates (notation $1\{y^{(i)} = 1\}$ returns 1 only when $y^{(i)} = 1$ and $|V|$ represents the size of the feature set):

$$\phi_{k|y=1} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 1\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} n_i + |V|} \quad (2)$$

$$\phi_{k|y=0} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 0\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} n_i + |V|} \quad (3)$$

$$\phi_k = \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}}{m} \quad (4)$$

The most predictive features are then the features that maximize this value:

$$\text{abs}(\log(\frac{\phi_{i|y=1}}{\phi_{i|y=0}})) \quad (5)$$

Note, without the absolute value, the log of the fraction of conditional probabilities will give an assessment of how well a particular feature does at predicting label 1. The absolute value gives the feature that is most predictive of either label.

For the second technique, I performed a principal component analysis for each feature set over the training dataset. The principal components are simply the eigenvectors of the covariance matrix of the training data given by:

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} x^{(i)^T} \quad (6)$$

for training data $x^{(i)}$ and sample size $m$. The associated eigenvalue for the principal component represents the relative variance of the dataset explained by the principal component. For feature reduction, I ordered the principal components by variance explained, and kept the principal components that accounted for an accumulated 90% variance in the training dataset. For the feature set representing the number of mutations in each of the 68 genes of interest, this reduced the feature space from 68 to 49. For the feature set representing the number of mutation effects in each of the 68 genes, this reduced the feature space from 272 to 138.

The final technique was a combination of the first two, wherein I performed a principal component analysis on the 25 most predictive features from each feature set. This generally further reduced the size of feature sets from 25 to $< 20$.

## III. Methods

### A. Support Vector Machine

The first binary classifier I trained was the support vector machine with radial basis kernel. The support vector machine maximizes the minimum distance between a separating hyperplane and data points in each class. Namely, the support vector machine solves the optimization problem (over $m$ samples in the training dataset):

$$\max_{\alpha} \quad W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j\langle x^{(i)}, x^{(j)}\rangle \quad (7)$$

$$s.t. \quad 0 \le \alpha_i \le C, i = 1, ..., m \quad (8)$$

$$\sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \quad (9)$$

The inner product in the maximization allows me to utilize kernels to map my feature sets to higher dimensional space. In initial testing, I found the radial basis kernel function performed best, so I continued utilizing it here:

$$K(x, z) = \exp(-\gamma||x - z||^2) \quad (10)$$

To implement the support vector machine, I relied on the python library sklearn [**?**]. Sklearn allows users to set the parameters $C$ and $\gamma$ (default $C = 1$ and $\gamma = 1/n$ with $n$ as the size of the feature set). Partitioning the 501 sample training set into a 451 training set and a 50 sample validation set, I performed a parameter sweep over values of $C$ and $\gamma$ and found the best validation set accuracy after training with the training set using parameters $C = 5$ and $\gamma = 0.04$ with the largest feature set ($n = 272$).

$$accuracy = \frac{true\ positive + true\ negative}{m} \quad (11)$$

In addition to training a support vector machine classifier using the full feature sets, I also attempted to further reduce over-fitting by implementing forward feature selection with 10-fold cross validation. Features that yielded the best 10-fold cross validation accuracy were iteratively selected until there was a decline in 10-fold cross validation accuracy with additional features. To perform 10-fold cross validation, the training dataset was randomly divided into 10 partitions. At each feature selection iteration, the support vector machine was trained using a subset of 9 partitions, with the last partition used to compute the cross validation accuracy (defined in equation 12). This was repeated 10 times (once for each partition as the cross validation set), and the cross validation accuracies were averaged to obtain the 10-fold cross validation accuracy. The feature subset with the best 10-fold cross validation accuracy was then used to retrained the support vector machine on the full training dataset. Once I obtained a feature set that yielded satisfactory performance on the peak 10-fold cross validation accuracy and training set accuracy, I then evaluated its performance on the 159 sample test set.

### B. 3-layer Neural Network

The second classifier I trained was a 3-layer neural network with a single, fully connected hidden layer. The input layer contained $n$ nodes with $n$ corresponding to the size of the feature set. I tuned the number of hidden layer nodes by training the neural network with the largest feature set ($n = 272$) on a 451 sample subset of the training dataset and evaluating the accuracy of on the remaining 50 samples. With this, I

found a hidden layer with 15 nodes gave the best validation accuracy. To demonstrate neural network overfitting, I also trained a neural network with 50 hidden layer nodes. Finally, the output layer had 2 nodes corresponding to the two classes. I used a sigmoid activation for the hidden layer and a softmax activation for the output layer.

While this was unnecessary for binary classification, I used the softmax activation so that I could easily extend the neural network to handle future multi-class classification of tumor stage. I used a mini-batch gradient descent with a cross-entropy loss function to train the neural network:

$$CE(y, \hat{y}) = -\sum_{k=1}^{2} y_k \log \hat{y}_k \tag{12}$$

$$J_{MB} = \frac{1}{B} \sum_{i=1}^{B} CE(y^{(i)}, \hat{y}^{(i)}) \tag{13}$$

where $y$ is the one-hot representation of the sample label and $J_{MB}$ is the cost function over the mini-batch. In addition to tuning for the number hidden layer nodes, I also ran a parameter sweep to identify the best batch size (50), number of epochs (500), and learning rate (0.1). The original 501 sample training dataset was randomly partitioned into a 451 sample neural network training set and a 50 sample development set. The neural network was trained using the 451 sample neural network training set, and evaluated against the 50 sample development set. Once I obtained a neural network with satisfactory performance in both the training and development sets, I evaluated its performance on the 159 sample test set.

### C. 4-layer Neural Network

The final classifier I developed was a 4-layer neural network (Figure 3) that was built to only be compatible with the feature set representing the number of mutation effects in each of the 68 genes ($n = 272$ features). The input layer is a 272 node layer with one node for each feature. The second layer contains 68 nodes with each node representing one of the genes. The input layer and second layer are not fully connected, but instead have connections only between nodes associated with the same gene. Thus, in this case, each node in the second layer connects to 4 nodes in the input layer (one node for each mutation effect in the gene represented by the second layer node). The third layer is then a fully connected hidden layer with 15 nodes and the output layer again contains 2 nodes. The second and third layers utilize sigmoid activations, with the output layer utilizing a softmax activation, and cost function being defined as in the 3-layer neural network.

Because the input and second layers are not fully connected, the parameter matrix representing these connections is sparse. To implement this sparse connectivity, I first treated the layers as fully connected (computing the gradients normally), and then zeroed out indices associated with connections that did not exist. This neural network was trained and evaluated in the same manner as the 3-layer neural network.

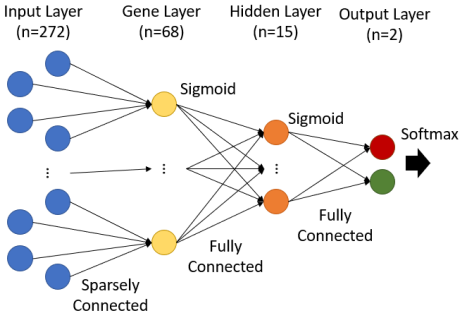


Fig. 3. Architecture of the 4-layer neural network. The input layer is sparsely connected with the second (gene) layer. Nodes in the second layer represent genes of interest, and thus only input layer nodes representing mutation effects in a gene are connected to the node representing that same gene in the second layer. The second layer uses a sigmoid activation and is fully connected with the third layer (15 nodes). The third layer uses a sigmoid activation and is fully connected to the output layer, which outputs a softmax activation.

## IV. Results and Discussion

Figure 4 displays aggregated training set and test set accuracies for the support vector machine and 3-layer neural network over all of the available feature sets and feature reduction techniques. Generally, the test set accuracies were between 55% and 60%, and training set accuracies occasionally exceeded 80%, particularly in cases where no feature reduction techniques were employed (support vector machine without forward feature selection, and the 50-node hidden layer neural network cases). This indicates substantial overfitting in these models due to the relatively large size of the feature or parameter space compared to the small datasets. While methods with feature reduction helped reduce overfitting (support vector machine classifier with forward feature selection and 15-node hidden layer neural network trained on principal components of naive-bayes selected features), the feature reduction methods only lowered the training accuracy rather than improving the test accuracy. This is likely because the feature reduction techniques removed important information that may have been helpful in classification.

The complex 4-layer neural network achieved a 69.4% test set accuracy with a 75.8% training set accuracy, a marked improvement over the other techniques. As the best performing classifier, I also computed an area under the receiver-operating curve (Figure 5) of 67.9%. The receiver-operator curve was created by shifting the decision boundary and comparing the softmax outputs to different decision boundaries. For each decision boundary, I computed the true positive rate ($TPR = TP/(TP + FN)$) and false positive rate ($FPR = FP/(FP + TN)$) with TP = number of true positives, FN = number of false negatives, FP = number of false positives, and TN = number of true negatives.

Despite using the largest feature set ($n = 272$), the neural network yielded good performance and little overfitting (similar training and test set accuracies). Unlike the feature reduction methods, my 4-layer neural network enforces relationships between mutations (input layer), the genes in

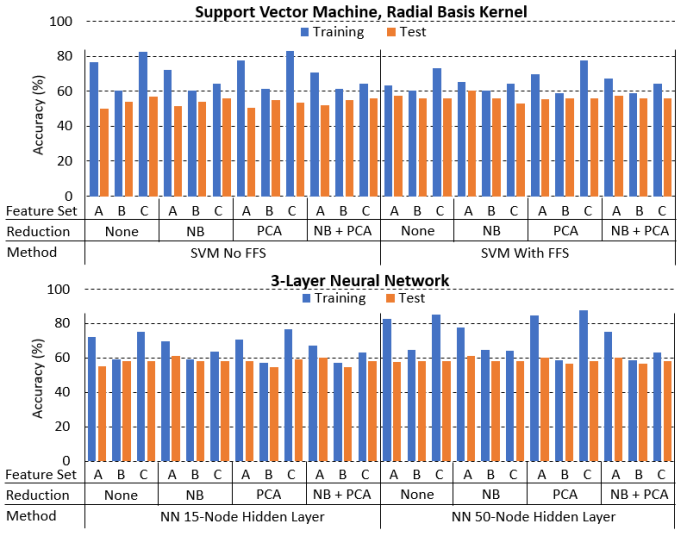


Fig. 4. Aggregated performance of the support vector machine and 3-layer neural network trained with the defined feature sets and feature reduction techniques. A refers to the feature set representing number of mutations in each gene, B refers to the feature set representing number of mutation effects total, and C refers to number of mutation effects in each gene. NB = Naive-Bayes feature reduction, PCA = principal component analysis, SVM = support vector machine, FFS = forward feature selection, and NN = neural network.

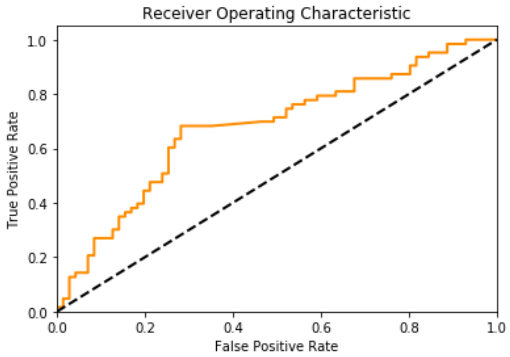


Fig. 5. ROC for the complex 4-layer neural network demonstrating its performance with varying decision boundaries. AUC = 67.9%

which they appear (gene 2), and interactions between genes (layer 3) that may be predictive of tumor stage (output layer). Effectively, this neural network architecture removes direct interactions between specific mutation effects in different genes, and instead forces those interactions to occur at the gene level rather than the mutation level.

Despite the poor performance on the test set for these classifiers, I can still make some inferences on the importance of various features. Figure 6 shows the principal components for the 25 genes whose mutation best predicted tumor stage. The first 19 principal components accounted for 90% of the variance in training data. The figure also displays which of the 25 genes and which of the 19 principal components were separately selected in forward feature selection with the support vector classifier, and the three circled blocks represent genes that were both selected directly, and significantly represented in a selected principal component. Two of the genes identified, ZMYND10 and SMARCA4, have been previously implicated as having a role in cancer occurrence [8], [12].

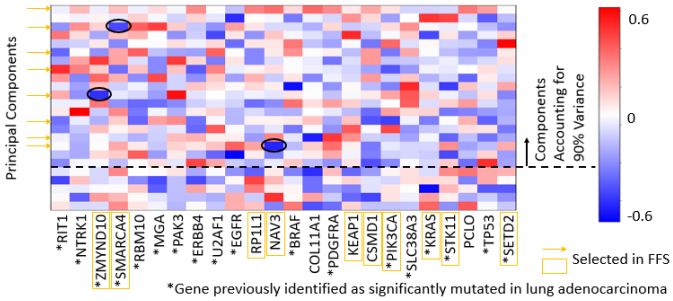


Fig. 6. Principal component analysis on most predictive genes from Naive-Bayes. Features selected from the support vector machine forward feature selection procedure indicate the importance of several genes, including ZMYND10, SMARCA4, and NAV.

From the trained weights in the neural network, we can again infer the importance of the different genes, and the types of mutations responsible for altering gene function in late vs. early stage tumors. Looking at the node in the output layer corresponding to early stage cancers, I found that the largest weight (-2.57) to this node comes from node 7 in the 15-node third layer. From here, two nodes in the gene layer representing the ZMYND10 and STK11 gene have large weights (1.82 and -2.65 respectively) to node 7. For the ZMYND10 gene, the highest weight (2.11) comes from the node representing a missense mutation in the ZMYND10 gene. For the STK11 gene, the highest weight (-2.56) comes from the node representing an insertion mutation in the STK11 gene. Both ZMYND10 and STK11 are thought to suppress tumors [11], and the weighting pattern in the 4-layer neural net implies that mutations within these genes are not predictive of early stage tumors. This is consistent with my previous support vector machine analysis and hypothesized gene functions.

## V. Conclusion

In conclusion, I developed a 4-layer neural network that can predict with 69.4% accuracy whether a lung adenocarcinoma tumor is localized to one lung. The structure of this neural network can be interpreted as relating which types of mutations result in substantial changes in gene function, and which genes, when altered, best predict tumor stage. Simpler support vector machine and 3-layer neural network classifiers suffered from over-fitting that could not be adequately corrected with feature reduction techniques. However, our classifiers tended to indicate the importance of certain genes regulating cell proliferation (ZMYND10) in predicting tumor stage.

In the future, I would like to explore techniques to account for the small publicly available datasets (bootstrapping or other data augmentation and resampling techniques). In addition, I would like to incorporate features discovered from the neural network to retrain a support vector machine, which I believe would be better at reducing the size of the feature set without losing significant information necessary for classification.

## REFERENCES


[1] D. G. Beer, S. L. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas *et al.*, "Gene-expression profiles predict survival of patients with lung adenocarcinoma." *Nature medicine*, vol. 8, no. 8, 2002.

[2] A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette *et al.*, "Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses," *Proceedings of the National Academy of Sciences*, vol. 98, no. 24, pp. 13 790–13 795, 2001.

[3] L. Ding, G. Getz, D. A. Wheeler, E. R. Mardis, M. D. McLellan, K. Cibulskis, C. Sougnez, H. Greulich, D. M. Muzny, M. B. Morgan *et al.*, "Somatic mutations affect key pathways in lung adenocarcinoma," *Nature*, vol. 455, no. 7216, pp. 1069–1075, 2008.

[4] R. Doll and A. B. Hill, "Lung cancer and other causes of death in relation to smoking," *British medical journal*, vol. 2, no. 5001, p. 1071, 1956.

[5] S. B. Edge and C. C. Compton, "The american joint committee on cancer: the 7th edition of the ajcc cancer staging manual and the future of tnm," *Annals of surgical oncology*, vol. 17, no. 6, pp. 1471–1474, 2010.

[6] J. Gao, B. A. Aksoy, U. Dogrusoz, G. Dresdner, B. Gross, S. O. Sumer, Y. Sun, A. Jacobsen, R. Sinha, E. Larsson *et al.*, "Integrative analysis of complex cancer genomics and clinical profiles using the cbioportal," *Science signaling*, vol. 6, no. 269, p. pl1, 2013.

[7] N. C. Institute, "Cancer stat facts: Lung and bronchus cancer," https://seer.cancer.gov/statfacts/html/lungb.html.

[8] P. P. Medina, O. A. Romero, T. Kohno, L. M. Montuenga, R. Pio, J. Yokota, and M. Sanchez-Cespedes, "Frequent brg1/smarca4–inactivating mutations in human lung cancer cell lines," *Human mutation*, vol. 29, no. 5, pp. 617–622, 2008.

[9] C. G. A. R. Network *et al.*, "Comprehensive molecular profiling of lung adenocarcinoma," *Nature*, vol. 511, no. 7511, pp. 543–550, 2014.

[10] G. P. Pfeifer, M. F. Denissenko, M. Olivier, N. Tretyakova, S. S. Hecht, and P. Hainaut, "Tobacco smoke carcinogens, dna damage and p53 mutations in smoking-associated cancers," *Oncogene*, vol. 21, no. 48, p. 7435, 2002.

[11] M. Sanchez-Cespedes, P. Parrella, M. Esteller, S. Nomoto, B. Trink, J. M. Engles, W. H. Westra, J. G. Herman, and D. Sidransky, "Inactivation of lkb1/stk11 is a common event in adenocarcinomas of the lung," *Cancer research*, vol. 62, no. 13, pp. 3659–3662, 2002.

[12] W. L. Yau, H. L. Lung, E. R. Zabarovsky, M. I. Lerman, J. S.-t. Sham, D. T.-t. Chua, S. W. Tsao, E. J. Stanbridge, and M. L. Lung, "Functional studies of the chromosome 3p21. 3 candidate tumor suppressor gene blu/zmynd10 in nasopharyngeal carcinoma," *International journal of cancer*, vol. 119, no. 12, pp. 2821–2826, 2006.