CS 365 Project Milestone

Jae Hong Lee and Wyatt Napier
Boston University
jhonglee@bu.edu wnapier@bu.edu

Abstract

For our project milestone we're using Principal-Component Analysis (PCA) and Support Vector Machines (SVM) to predict housing prices based off of this dataset provided by Kaggle.

1 Principal Component Analysis

Principal Component Analysis (PCA) is a method of compressing the dimension of data to make learning easier and prediction more efficient. We want to go from representing the data in R^d to a significantly smaller space R^p by using p principal components from the eigendecomposition.

We want to approximate an input vector $\mathbf{x^{(n)}} \in R^d$ with basis vectors $\mathbf{v_1}, ..., \mathbf{v_d} \in R^d$ by fulfilling the following equation where each α is a representation coefficient: $\mathbf{x^{(n)}} \approx \sum_{i=1}^{p} \alpha_i \mathbf{v_i}$. To do so we need to find both the basis vectors $\{\mathbf{v_i}\}_{\mathbf{i=1}}^{\mathbf{p}}$ and coefficients $\{\alpha_i\}_{i=1}^{p}$. We do this using an eigendecomposition which is often derived from SVD or the definition of eignevalues and eigenvectors: $\mathbf{Au} = \lambda\mathbf{u} \rightarrow \mathbf{A} = \mathbf{U\Lambda U^{-1}}$.

For just one row of data we would need to solve this optimization problem: $(\hat{\alpha}, \hat{\mathbf{v}}) = argmin_{||\mathbf{v}||_2=1,\alpha}||\mathbf{x} - \alpha\mathbf{v}||^2$. Basically what this means is we want to find the coefficient and the basis vector such that we can minimize the distance of $\mathbf{x}$ from the scaled vector $\alpha\mathbf{v}$. To generalize this to the entire matrix we have the following equations:

$$\hat{\mathbf{v}} = argmin_{||\mathbf{v}||_2=1} E||\mathbf{X} - \alpha\mathbf{v}||^2 = argmax_{||\mathbf{v}||_2=1} \mathbf{v^T \Sigma v}$$

We want to maximize the variance of the data, while also minimizing the distance of the data from the basis vectors. These two problems happen to actually be the same problem. Since $\mathbf{\Sigma}$ is just the covariance matrix from $\mathbf{\Sigma} = E[\mathbf{X^T X}]$, we know that the $\hat{\mathbf{v}}$ has a solution $\mathbf{u_1}$ which is the first column of the eignevector matrix $\mathbf{U}$ assuming it is ordered by magnitude of corresponding eigenvalue.

Stepping back for a moment, we want to maximize the variance along the basis vectors which will also minimize the distance of each point from that line. Since we're using the covariance matrix the diagonal will just be the variance. When we do the eigendecomposition, the eigenvalues are then a measure of the variance so the magnitude of the eigenvalues corresponds to the variance of each.

To clarify, the eigendecomposition of the dxd matrix $\mathbf{\Sigma}$ will give us the eigenvector matrix $\mathbf{U}$ and the eigenvalue matrix $\mathbf{S}$. As part of PCA we want to use just p principle components (started with d components) so we just truncate $\mathbf{U}$ and $\mathbf{S}$ to use only the first p eigenvectors and eigenvalues. In doing so, we project the points $\{\mathbf{x^{(1)}}, ..., \mathbf{x^{(n)}}\}$ onto the space spanned by the first p eigenvectors of $\mathbf{U}$. We use the following equation for a single point and repeat for the entire dataset to compress it:

$$\mathbf{U_p^T x^{(n)}} = \alpha^{(\mathbf{n})}$$

Overall, the structure of PCA is to center the mean of the data on the origin and then find the eigendecomposition of the covariance matrix so that we can then do the reduction. In some applications the data is first scaled for each feature before applying SVD. PCA'a foundations lay in simple linear algebraic techniques such as SVD and eigendecomposition and as such it is a true testament to the power of linear algebra.

Sources:

CS132 Textbook - Mark Crovella
Probability for Data Science - Steven Chan

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

2 Support Vector Machine

2.1 Decision Rule

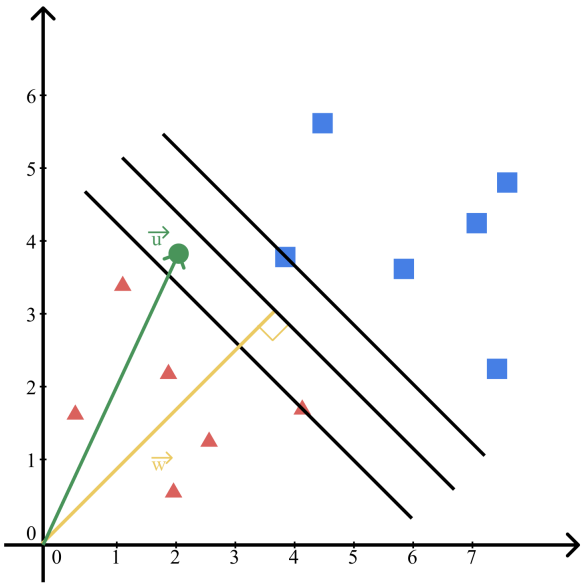

Figure 1: Decision Boundary

The space in the middle is drawn based on the widest path between the closest vectors (commonly called support vectors) of two different categories. Therefore, it's referred to as the widest street approach. In the method of dividing examples of two kinds: triangles and rectangles, this path is made as wide as possible. There is a vector $\vec{w}$ of some length. This $\vec{w}$ is perpendicular to the dividing center-line of the plane, and therefore the distance from the origin to the center-line is minimal. The distance of this vector is unknown.

And there is another unknown point. It is represented by $\vec{u}$ in the figure. It is important to determine whether this point is closer to a triangle or a rectangle. Now, by projecting this point onto the $\vec{w}$ vector, we can determine how many times it is proportional to the $\vec{w}$ vector. If the length of the projection of $\vec{u}$ onto $\vec{w}$ is closer to the center-line, it is classified as a triangle; otherwise, if it is further than the center-line, it is classified as a rectangle. The figure above $\vec{u}$ would be classified as a triangle.

The projection of $\vec{u}$ to $\vec{w}$ is the dot product of two vectors.

This is SVM's Decision Rule

$$\vec{w} \cdot \vec{u} \geq \textit{some constant } c$$
$$\textit{if } c = -b,\ \vec{w} \cdot \vec{u} + b \geq 0 \textit{ then Rectangle}$$

2.2 Margin Constraints

What conditions are needed for the constants b and vector $\vec{w}$? We only know that vector $\vec{w}$ is perpendicular to the central line. However, since $\vec{w}$ can be perpendicular to the central line regardless of its length, there are no constraints on $\vec{w}$ and b. To determine $\vec{w}$ and b, some constraints are required.

Using Figure 1 as an example, to determine $\vec{w}$ and b's values, for rectangular vectors which is farther than the center-line, find two variables $\vec{w}$ and b that satisfy the equation $\vec{w} \cdot \vec{x_{rec}} + b \geq 1$. The output will be greater than 1 for all rectangle vectors. In addition, the previously found $\vec{w}$ and b should satisfy the equation $\vec{w} \cdot \vec{x_{tri}} + b \leq -1$ for triangle vectors.

However, finding the values of $\vec{w}$ and b using these two equations is not an easy task. For mathematical convenience, let's define a variable y_i which returns a value of +1 for rectangle vectors and -1 for triangle vectors. Therefore, for the equation of the rectangle, $\vec{w} \cdot \vec{x_{rec}} + b \geq 1$ the expression involving y_i: $y_i \cdot (\vec{w} \cdot \vec{x_{rec}} + b) \geq 1$ and for the triangle: $y_i \cdot (\vec{w} \cdot \vec{x_{tri}} + b) \geq 1$. They both have the same equations $y_i \cdot (\vec{w} \cdot \vec{x_{rec\ or\ tri}} + b) - 1 \geq 0$. In a special case, for the vectors inside the widest street area (gutter) the equation holds $y_i \cdot (\vec{w} \cdot \vec{x_{rec\ or\ tri}} + b) - 1 = 0$.

2.3 Get the width of the street

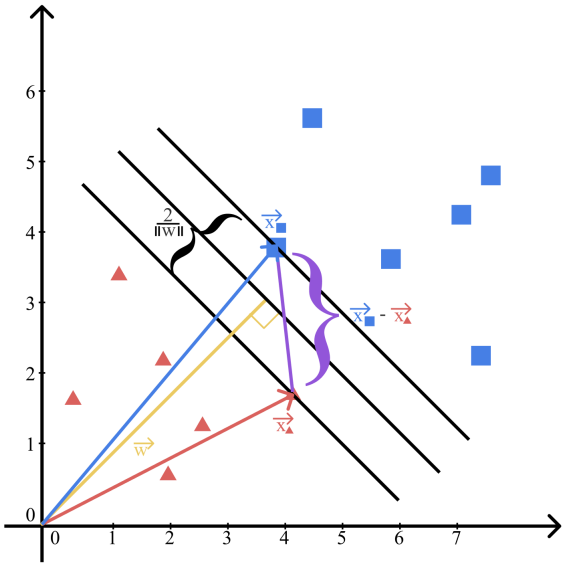

Figure 2: Width of the street

What is the widest path to divide triangles and rectangles? Despite not knowing the path widths, we can calculate the difference between the two support vectors (the closest two vectors to the gutter) of the triangles and rectangles. Using these two support vectors as $\vec{x_{rec}}$ and $\vec{x_{tri}}$, respectively, we can calculate the difference between the two vectors as $(\vec{x_{rec}} - \vec{x_{tri}})$. We can determine the width of the street by taking the dot product of the difference between these two vectors and the unit normal to the center-line. In other words, if we have a unit vector pointing in the direction of the center-line, we can calculate the street width.

$$width \ = \ (\vec{x_{rec}} - \vec{x_{tri}}) \cdot \frac{\vec{w}}{\|w\|}$$

From the previous margin constraints, we have the equation, $y_i \cdot (\vec{w} \cdot \vec{x}_{rec \ or \ tri} + b) - 1 = 0$. If we have the rectangle support vector, y_{rec} will be 1 and rest of the equation will be $(\vec{w} \cdot \vec{x_{rec}} + b) - 1 = 0$. So $\vec{x_{rec}} \cdot \vec{w}$ will be $1 - b$. Similarly, when we have the triangle support vector, y_{tri} will be -1 and $\vec{x_{tri}} \cdot \vec{w}$ will be $-1 - b$. With this equations we got $\vec{x_{rec}} = 1 - b$ and $\vec{x_{tri}} = -1 - b$. So the width is $(\vec{x_{rec}} - \vec{x_{tri}}) \cdot \frac{\vec{w}}{\|w\|} = \frac{\vec{x_{rec}}\vec{w} - \vec{x_{tri}}\vec{w}}{\|w\|} = \frac{1-b-(-1-b)}{\|w\|} = \frac{2}{\|w\|}$.

We got the width: $\frac{2}{\|w\|}$. We want to maximize the width. For mathematically convenience, it is okay to maximize $\frac{1}{\|w\|}$. Which also leads to minimize $\|w\|$. Eventually it is minimizing $\frac{1}{2} \cdot \|w\|^2$.

2.4 Maximize Width

From previous section, we learn in order to maximize the width of the street, need to minimize $\frac{1}{2} \cdot \|w\|^2$. To find the min $\|w\|$ we use Lagrange multiplier.

Applying Lagrange Multiplier, $L = \frac{1}{2} \|w\|^2 - \sum \alpha_i [y_i(\vec{w} \cdot \vec{x_i} + b) - 1]$.

Then, in order to find the minimum, derivative the equation L and find w, b which makes 0: $\frac{dL}{dw} = \vec{w} - \sum \alpha_i y_i \vec{x_i} \rightarrow 0 \rightarrow \vec{w} = \sum \alpha_i y_i \vec{x_i}$ $\vec{w}$ is the linear sum of these vectors α_i, y_i, $\vec{x_i}$. Also, $\frac{dL}{db} = -\sum \alpha_i y_i => 0$.

Then plug back in new derivative derivative to the original Lagrange Multiplier:
$L = \frac{1}{2} \sum (\alpha_i y_i \vec{x_i}) \cdot \sum (\alpha_j y_j \vec{x_j}) - \sum (\alpha_i y_i \vec{x_i}) \sum (\alpha_j y_j \vec{x_j}) - \sum \alpha_i y_i b + \sum \alpha_i$

Then, we know b is constant so takes it outside of sum.
$L = \frac{1}{2} \sum (\alpha_i y_i \vec{x_i}) \cdot \sum (\alpha_j y_j \vec{x_j}) - \sum (\alpha_i y_i \vec{x_i}) \sum (\alpha_j y_j \vec{x_j}) - b \sum \alpha_i y_i + \sum \alpha_i$

From derivative of b, $-\sum \alpha_i y_i = 0$.

$L = \frac{1}{2}\sum(\alpha_i y_i \vec{x_i}) \cdot \sum(\alpha_j y_j \vec{x_j}) - \sum(\alpha_i y_i \vec{x_i}) \sum(\alpha_j y_j \vec{x_j}) + \sum \alpha_i = \sum \alpha_i - \frac{1}{2}\sum(\alpha_i y_i \vec{x_i}) \cdot \sum(\alpha_j y_j \vec{x_j}) = \sum \alpha_i - \frac{1}{2}\sum\sum a_i a_j y_i y_j \cdot x_i \cdot x_j$

To summarize, we see that the optimization is only depends on the pair of samples:

$$L = \sum \alpha_i - \frac{1}{2}\sum\sum a_i a_j y_i y_j \cdot x_i \cdot x_j$$

2.5 Apply back to Decision Rule

Now plug w back in decision rule from the figure 1.

$$\sum(a_i y_i \vec{x_i}) \cdot \vec{u} + b \geq 0, \ then \ Rectangle$$

Sources

Support Vector Machines - MIT OpenCourseWare

Support Vector Machine: All You Need to Know! - Intuitive Machine Learning

3 Experiments

3.1 Set Up

Colab Link : Google Colab

Housing Dataset Link : Google Kaggle

Based on our research on PCA and SVM methods, we applied Support Vector Regression (SVR) to our housing price dataset, aiming to predict house prices using 80 features. The dataset contains a large number of features, so we used PCA to reduce the number of feature columns to 10.

To apply PCA, we first scaled our data so that the eigendecomposition wasn't skewed and then we used sklearn's PCA method to reduce the dimensionality of our dataset to 10 columns by projecting the data onto the space spanned by these 10 eigenvectors. We then centered these results around the origin by transforming the data.

3.2 Result

We applied two different kernels to SVR: the Linear Kernel and the RBF Kernel. Based on the results of the experiments, R-squared score are as follow:

Linear SVR R-squared score was: 0.8519180665477

RBF SVR R-squared score was: -0.05160640422659202

As observed, using linear kernel generated better results. We used the R-squared score as the scoring function. $R^2 = 1 - \frac{u}{v}$, where $u = \sum(y_{true} - y_{predict})^2$ and $v = \sum(y_{true} - y_{true\ mean})^2$. A value closer to 1 indicates a better model fit.

Hypothesis to the result: Based on comprehensive analysis, the linear kernel performed better than the RBF kernel due to a reduced subset generated by PCA, which benefited the linear kernel. For our next report, we will study kernels more thoroughly and implement the kernel method that is most suitable for our dataset.

4 Next Steps

One of the most important steps we need to take is optimizing our pre-processing such that we better account for the NA's in largely numerical columns of the data such as 'LotFrontage', 'GarageYrBlt', and 'MasVnrArea'. We have a few options such as deleting those columns, or deleting the associated rows with NA's or applying one-hot encoding.

Additionally, for both PCA and SVM, we need to learn about the kernel so that we can apply it to try to optimize our model. Depending on the distribution of our data, like if it is nonlinear, I believe that the kernel would dramatically increase the accuracy of our model as was hinted at in the end of the MIT SVM lecture.