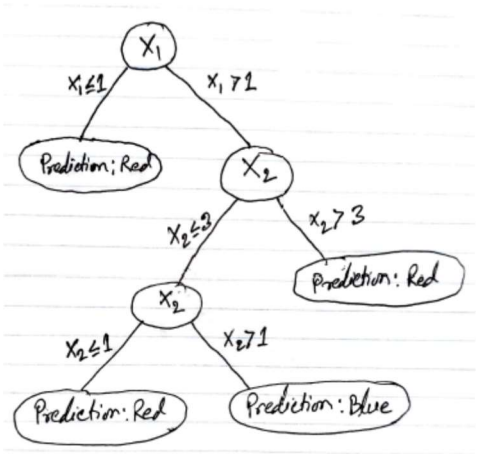
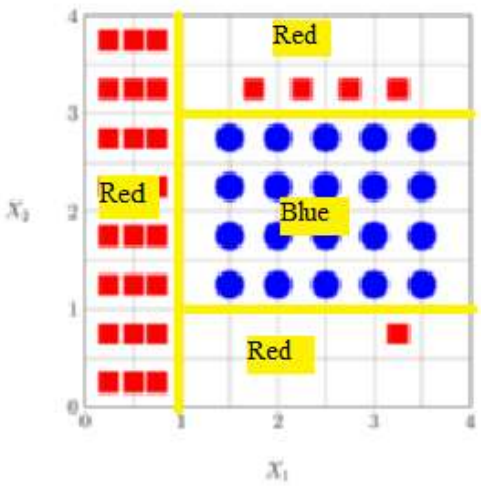
Problem 1:

A. **Decision Tree**:
The objective of construction of a decision tree to subset data into smaller group with similar characteristics of features. It starts with whole data and each step subset data into groups like a tree. For example, in this problem if we split $x_1 \leq 1$ and $x_1 > 1$, we can easily classify a large portion of red. Similarly, if we continue splitting based on other variable, we will be able to group then into smaller subset of data with similar characteristics. To take decision about how we can split, we can use different splitting technique like Gini index, entropy etc.

B.



C.



Problem 2:

A. If $\lambda = 0$, we will get the model that may over-fit data. That is we will get

$$\hat{Y} = 2.1 - 0.4X_1 + 1.3X_2 + 3.8X_3 - 0.9X_4 + 0.5X_5$$

If $\lambda \to \infty$, we will get the model that may under-fit data. That is, we will get the mean of Y

$$\frac{1}{100}\sum_{i=1}^{100} Y_i = 3.8$$

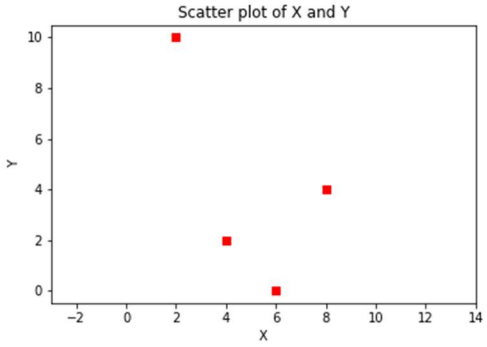
B.

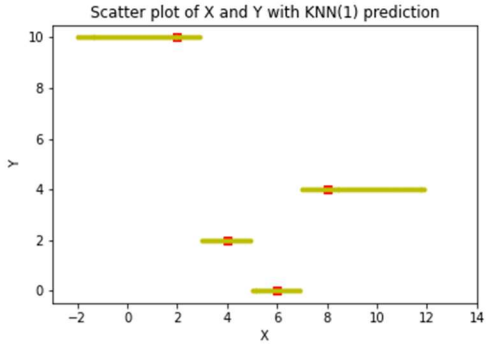
1. The penalty shrinks all the coefficients towards zero. Because we try to minimize loss function (1) and if coefficients become large, due to penalty the loss function will be larger. Loss function will be smaller when the penalty that is all a_i's are close to zero. It works as a tread-off between bias-variance. As λ increases, model complexity decreases as a result variance decreases and bias increases.
2. $2^5 = 32$ optimization calls needed. (Though I am not sure what optimization call means here). If $p = 0$ means L_0 norm ($|.|_0$) not the power of $|.|^0$, we normally use best subset search. Based on that we need 32 optimization calls.
3. Most used methods for finding best model are forward and backward stepwise selection. In forward search we start a model containing no feature with only intercept, and then include features one by one until all features includes in the model. Similarly, in backward search we start a model containing all features, and then exclude features one by one until no features are in the model except intercept. Both these procedures need $1 + \frac{k(k+1)}{2}$ features where $k = 5$ represents number of features. Out of these all fitted models, we select the best (based on AIC, BIC or MSE on test data etc.) model.

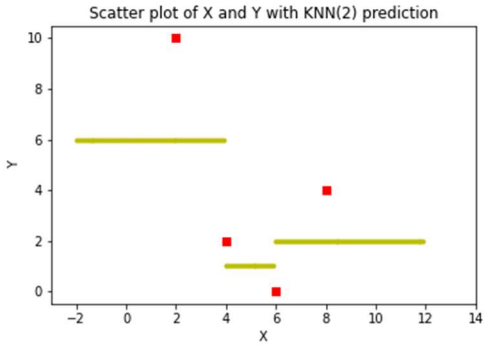
C. If we use $p = 1$ or $p = 2$, for a fixed λ, we just need one optimization call to solve (1).

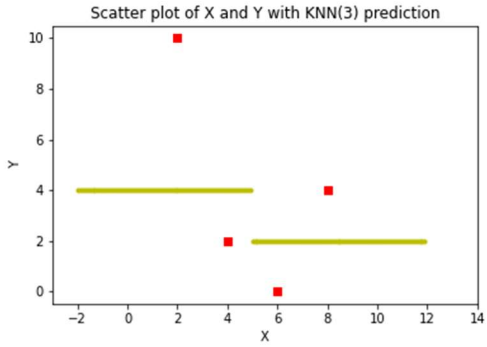
D. With $p = 2$, above loss function will include all 5 features in the final model even with large λ though the penalty will shrink all the coefficients towards zero, but it will not set any of them exactly to zero unless $\lambda = \infty$. However, for $p = 1$ some coefficient estimates may be exactly equal to zero. For $\lambda = 0$ both $p = 1$ and $p = 2$ will produce least square regression estimate for all 5 features. $p = 1$ minimizes loss function within a rectangular area of coefficient values and $p = 2$ minimizes loss function within a sphere of coefficient values.

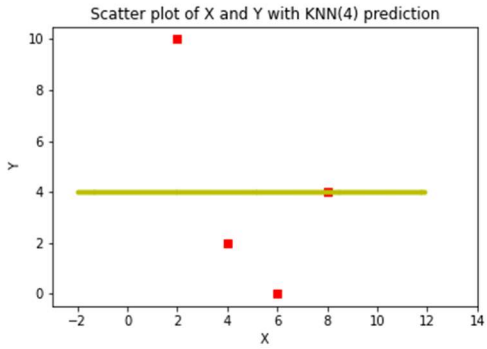
Problem 3:











Python code for above plots:

```
from matplotlib import pyplot as plt
from sklearn.neighbors import KNeighborsRegressor as kn
import numpy as np
x_train = np.array([2,4,6,8])
y_train = np.array([10,2,0,4])
plt.scatter(x_train,y_train,color = 'r', marker='s')
plt.title('Scatter plot of X and Y')
plt.xlim(-3,14)
plt.xlabel('X')
plt.ylabel('Y')
plt.savefig('plot.png')
plt.show()
x_test = np.arange(-2, 12, 0.1)
```

```
for i in [1,2,3,4]:
    model_kn = kn(n_neighbors=i, weights='uniform',
                  algorithm='auto').fit(x_train.reshape(-1, 1), y_train)
    Z = model_kn.predict(x_test.reshape(-1, 1))
    plt.scatter(x_train,y_train,color = 'r', marker='s')
    plt.scatter(x_test, Z, color = 'y',marker='.')
    plt.title('Scatter plot of X and Y with KNN('+str(i)+') prediction')
    plt.xlim(-3,14)
    plt.xlabel('X')
    plt.ylabel('Y')
    plt.savefig(str(i)+'plot.png')
    plt.show()
```

For this example, $KNN(k)$ finds the k nearest points of X and take the average of corresponding Y as the prediction for given X. Following table show the calculation

KNN(k)	X Range	X	Y	$\hat{Y}$	Average of
1	X <=3	2	10	10	10
	X>3 & X<=5	4	2	2	2
	X>5 & X<=7	6	0	0	0
	X>7	8	4	4	4
2	X<=5	2	10	6	10, 2
	X<=5	4	2	6	10, 2
	X>5 & X<=7	6	0	1	2, 0
	X>7	8	4	2	0, 4
3	X<=7	2	10	4	10, 2, 0
	X<=7	4	2	4	10, 2, 0
	X<=7	6	0	4	10, 2, 0
	X>7	8	4	2	2,0,4
4	any range of X	2	10	4	10, 2, 0, 4
	any range of X	4	2	4	10, 2, 0, 4
	any range of X	6	0	4	10, 2, 0, 4
	any range of X	8	4	4	10, 2, 0, 4

Problem 4:

Under-fitting: If both training and test data loss function measure (MSE) are very high (low for accuracy), we call it as under-fitting. It means that it is fitting the training and test data poorly.

Over-fitting: If the training data loss function measures (MSE) are very low (high for accuracy) compare to test/validation data loss function measure, we call it as over-fitting. It means that the model is fitting the training data well but fails to do so for the validation/test data.

Why both phenomena are undesirable: Both phenomena are not desirable for any model. Because out objective to find a model that will give us almost similar performance on training and test data. If we get a model that works well on training data but perform poorly on new data, will

not serve our purpose to predict based on that model. On the other hand, if any model perform poorly on both train and test will not also help us to use it.

How we can identify when they happen: Using training and validation/test data loss function we can identify over and under fitting. If there is any evidence that in the training data, we have very good performance but poor performance on test data indicates over-fitting. If both training and test performance are poor indicates under-fitting.

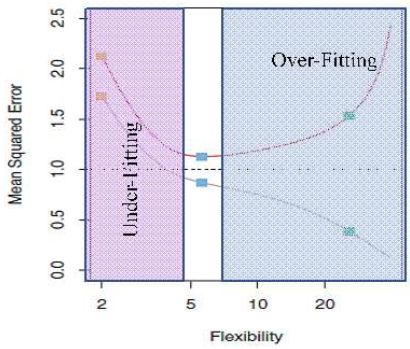


Figure: Lines represent train (black) and test (red) MSE with model complexity. Approximate shaded region represents over and under fitting.

Problem 5:

PCA is an unsupervised learning method mainly use to reduce dimension of data. It can be used to visualize and explore data with high dimension. PCA generates orthogonal linear combinations of features to create new variables. It can not be used as prediction model because it does not incorporate response variable. We can use PCA within other model like regression, SVM or any other supervised model as data reduction technique.

In linear regression we have a response variable and we try to find a linear relation of features with the response variable. However, in PCA, we do not use any response variable. We try to find some orthogonal linear combination of features only. This linear combination does not explain the relation of features with the response variable.

Problem 6:

A. Based on LDA classifier on this data, the boundary would be at $X_1 = 0$ for any value of X_2. Because LDA generate linear decision boundary and it will create the decision boundary at the middle of the mean of $\boldsymbol{X}$ of two classes. Since for both classes, the mean of X_2 is zero, implies that there is no distinction of mean of X_2 to any of the two classes. Based on only X_1 we can separate the two classes.
B. If we use QDA classifier on this data, we will get the same decision boundary as LDA. Because LDA considers equal covariance matrix for both classes but QDA can take unequal covariance matrix. However, if covariance matrix for both classes are same then QDA is equivalent to LDA. In this problem, we have equal covariance matrixes which implies that QDA and LDA are same for this data.

C. If we fit a maximal margin linear SVM classifier with no mistake allowed, it will create two vertical lines considered as margins at $X_1 = 1$ and $X_1 = -5$ and the boundary will be at $X_1 = -2$ where any value with $X_1 < -2$ will be predicted as -1 and any value with $X_1 \geq -2$ will be predicted as $+1$. Because linear SVM will find the least value of $X_1 = 1$ since all values of $X_2 = 0$ from $+1$ class and use it as the upper margin and lowest value of $X_1 = -5$ since all values of $X_2 = 0$ from -1 class and use it as lower margin. Taking average of these two margins will create the boundary for the classification this is the equal distance from upper and lower margins. Note that the outlier and all $X_1 = -5$ are called the support vectors.

D. If $C = 0$, we will get the same output as in previous part (C). It is called the maximal margin linear SVM classifier. If we consider $C > 0$ which is called soft margin linear SVM classifier means that no more than C observations can be on the wrong side of the hyperplane because $C = \sum_{i=1}^{2000} \epsilon_i$. That is, if $C > 1$, the outlier observation will be fall into the error margin and the margins and support vectors will be at $X_1 = -5$ and $X_1 = 5$ and the boundary will be at $X_1 = 0$ same as LDA and QDA.