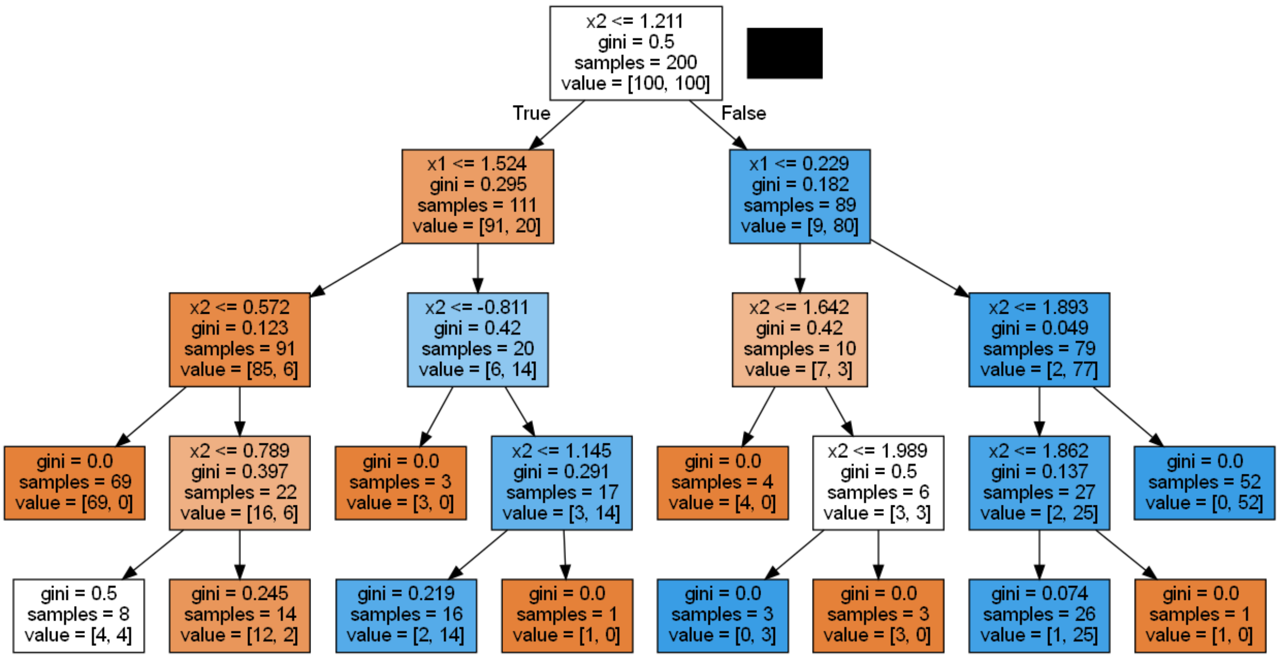
Problem 2

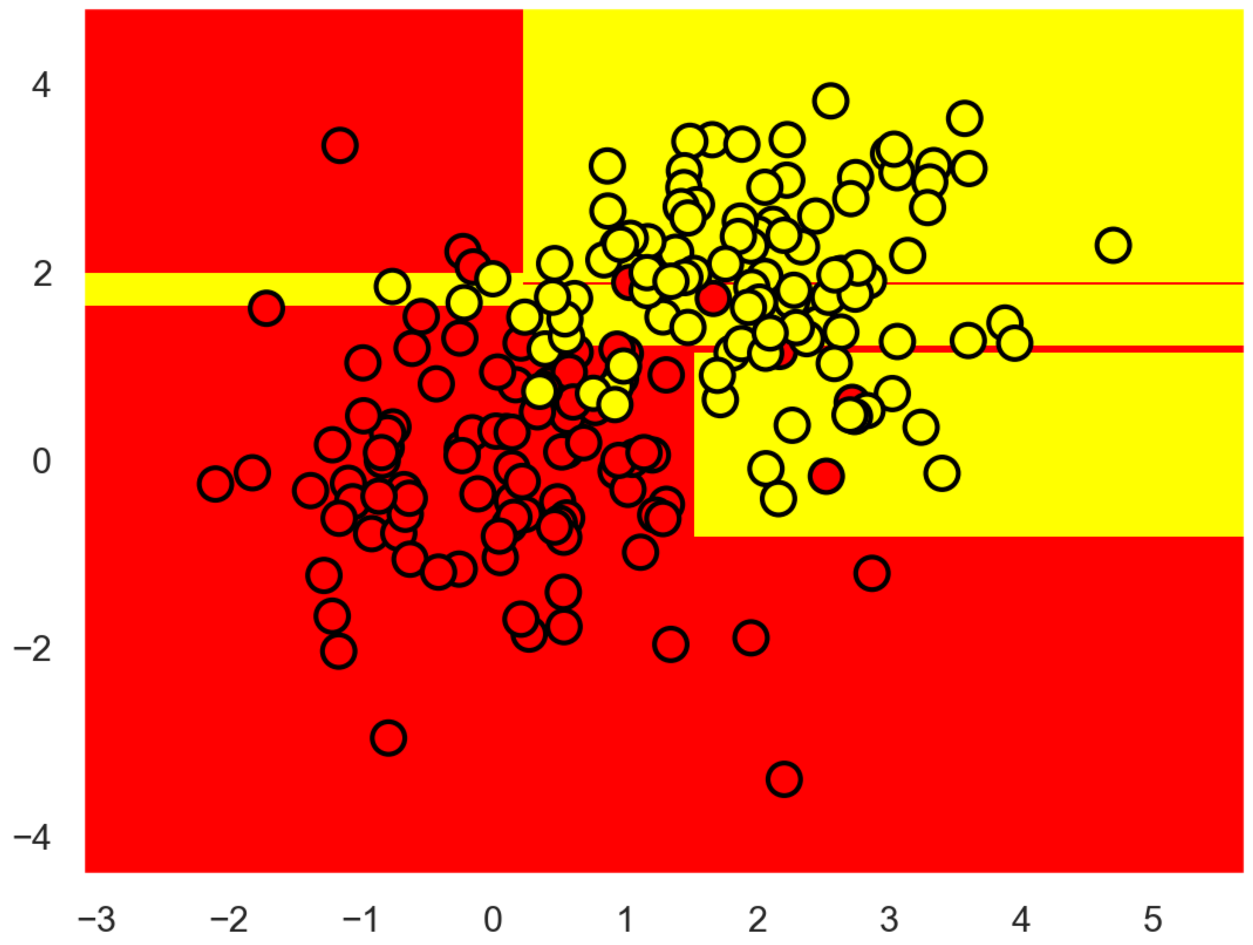
In this experiment, we will apply and visualize decision trees, apply kNN, finetune parameters, and learn about k-fold cross-validation. To visualize the decision tree, we need to install additional packages: Graphviz and pydotplus. Answer the following questions:

1. Decision tree classifier sklearn.tree.DecisionTreeClassifier has the parameter "max_depth", which defines the maximum depth of the tree, and "criterion", which measures the quality of the split. What happens if we don't specify any value for both parameters?
   The default criterian is "gini", the default max_depth=None. The function to measure the quality of a split is “gini” for the Gini impurity. Since max_depth=None, nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

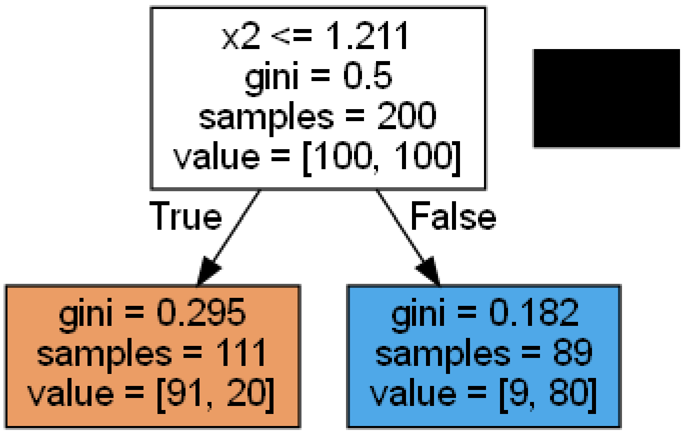
2. For the synthetic dataset, we separate two classes by training a decision tree. What does the boundary look like when we overfit (max_depth ≥ 4) and underfit (max_depth = 1) the decision tree on data? For both cases, paste the decision tree and the decision boundary from the Jupyter notebook output.
   When we overfit (max_depth ≥ 4) the decision tree on data, the boundary is very complicated, we can see that the tree "cuts" the space into many little rectangles for max_depth=4.



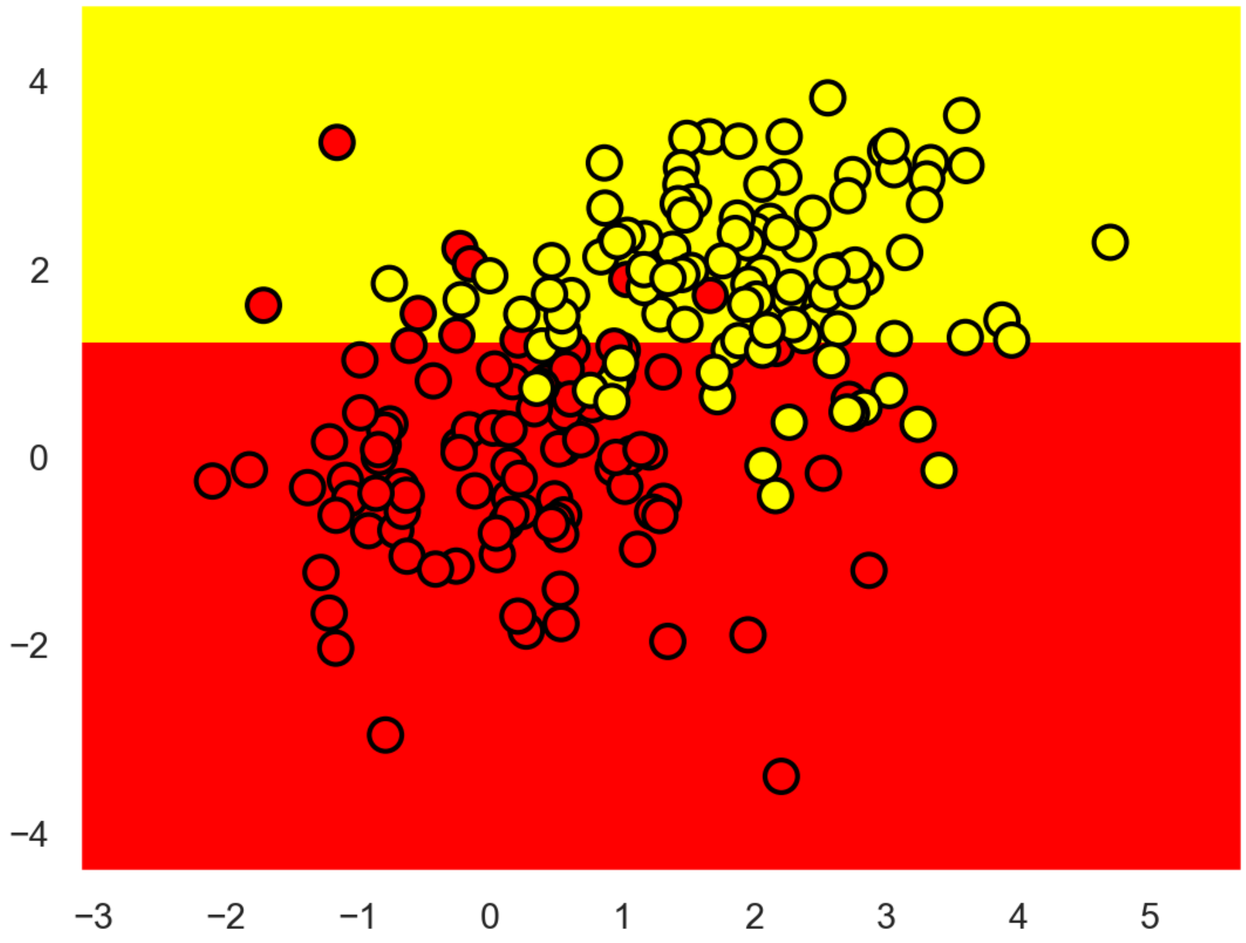
[redacted]



When we underfit (max_depth = 1) the decision tree on data, there is only one straight line serving as the boundary, which mis-classify many points.



[redacted]



3. For Bank Dataset, what are the 5 different age values that the decision tree used to construct the tree? What is the significance of these 5 values?
   The tree used the following 5 values to evaluate by age: 43.5, 19, 22.5, 30 and 32 years. These are exactly the mean values between the ages at which the target class "switches" from 1 to 0 or 0 to 1. There will be too many binary attributes to select from at each step during tree construction. Here, we check only those thresholds where the value of the target variable changes.

4. Given a dataset d, with n samples and m continuous features, what does Standard Scaler sklearn.preprocessing.StandardScaler do? Given dataset d = [[0, 0], [0, 0], [1, 1], [1, 1]], write down its scaler transformation.

   The Standard Scaler will standardize features by removing the mean and scaling to unit variance. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set.

Simply speaking, we do not want the "total day minutes" feature, which is on the order of hundreds, to affect the distance more than "total intl calls", which is generally less than 100.
The scaler transformation for d: [[-1., -1.], [-1., -1.], [1., 1.], [1., 1.]]

5. In the section Underfitting and Overfitting (Jupyter notebook), we have classified two datasets (a small one and a big one) using two different decision trees to demonstrate underfitting and overfitting. Briefly describe the experiments performed and what you learn from these experiments.

    - Briefly describe the experiments performed what?
        - It first defines two trees with different max_depth. For small_tree, its max_depth=2; For big_tree, its max_depth=8.
        - Then it describe the shapes of each data set. small_data_train has 233 instances with 18 feathures. big_data_train has 2332 instances with 18 feathures.
        - Next, it trains four trees (small_tree with small/big data set, big_tree with small/big data set)
        - Then it gets prediction results from the X_holdout, and the small_data_train.
        - At last, it uses the prediction results to calulate *train accuracy* and *test accuracy*.
    - what you learn from these experiments?
        - Overfitting: Bigger trees with more depth may have a higher accuracy on the training data set, but its performance compromises a lot on the testing set.
        - Underfitting: Smaller trees's accuracy for the training and testing set might not differ as much as bigger tree's, however, both accuracy are not very high.
        - As the size of the training set increases, bigger tree's performance improves, while, small tree's performance remains relatively poor.

6. In section Imbalanced Class (Jupyter notebook) , we have trained a couple of classifiers on the balanced and unbalanced datasets and evaluated their accuracy on balanced and unbalanced datasets. Furthermore, we have printed the confusion matrix. Briefly describe the experiments performed and what you learn from these experiments. Also, write down the experiments' precision, recall, and f1 score.

    - Briefly describe the experiments performed what?
        - write some helper functions
        - It first defines a tree which max_depth=None, seperates the pos and neg cases in both training and testing set.
        - Then it creates the balance_train, imbalance_train, balance_test, and imbalance_test sets, and describe the shape of each sets.
        - Next, it trains a tree on the balanced set, gets its train accuracy (1.0), prints the confusion matrix, and test this tree on balanced and imbalanced testing sets seperately.
        - Next, it trains a tree on the imbalanced set, gets its train accuracy (1.0), prints the confusion matrix, and test this tree on balanced and imbalanced testing sets seperately.
        - and another tree on the imbalanced set.
    - what you learn from these experiments?
        - The accuracy measure is not suitable for handling data sets with imbalanced class distributions as it tends to favor classifiers that correctly classify the majority class. For example, test accuracy on *imbalanced data* for a decision tree trained on *balanced data* is **0.79**, which is not very low, however, its **f1_score = 0.05, recall = 0.02**.

        - A classifier that has a high precision is likely to have most of its positive predictions correct; A classifier with a high recall has a high chance of correctly identifying the positive instances of the data; F1 considers both the recall and precision.
        - A decision tree trained on balanced set, results in poor precision when it was tested against an imbalanced set; A decision tree trained on imbalanced set, results in poor recall when it was tested against an balanced set. A decision tree trained and test on an imbalanced set may have poor precision and recall.
    - experiments' precision, recall, and f1 score.
        - Train on balance and test on imbalance
            - Train [f1_score, precision, recall] for decision tree trained on balanced data is (1.0, 1.0, 1.0)
            - while [f1_score, precision, recall] on imbalanced data is (0.05291005291005291, 0.02717391304347826, 1.0)
        - Train on balance and test on balance
            - Train [f1_score, precision, recall] for decision tree trained on balanced data is (1.0, 1.0, 1.0)
            - while test [f1_score, precision, recall] on balanced data is (0.7892720306513409, 0.8046875, 0.7744360902255639)
        - Train on imbalance and test on balance
            - Train [f1_score, precision, recall] for decision tree trained on imbalanced data is (1.0, 1.0, 1.0)
            - while test [f1_score, precision, recall] on balanced data is (0.2727272727272727, 1.0, 0.15789473684210525)
        - Train on imbalance and test on imbalance
            - Train [f1_score, precision, recall] for decision tree trained on imbalanced data is (1.0, 1.0, 1.0)
            - while test [f1_score, precision, recall] on imbalanced data is (0.0, 0.0, 0.0)

7. In the section Adding irrelevant attributes (Jupyter notebook), we have added an irrelevant attribute to the dataset and have trained a decision tree classifier on it. Based on the test set results, what do you think has happened, and can you connect it with the class material? Briefly describe the experiment done and the intuition developed from these experiments.

    - What do you think has happened? Intuition behind the experiement? Accuracy of decision tree on test dataset dropped after training a tree on noised data set. Noise is the random component of a measurement error. It typically involves the distortion of a value or the addition of spurious objects. It could diminish a classfier's performance.
    - Briefly describe the experiment
        - First, it trains a tree with a normal set, and calculates its train and test accuracy.
        - Then, it add irrelevent attributes to the normal set to make it has noise.
        - Then, it trains a tree with a noised set, and alculates its train and test accuracy.

8. For the customer churn prediction task, we show that the accuracy of the decision tree is 94% when max_depth is set to 5. What happens to accuracy when we leave the value of max_depth at its default value? Explain the rise/fall of accuracy. When we leave the value of max_depth at its default value, accuracy for decision tree dropped to 0.92

This is due to overfitting. Since max_depth=None, nodes are expanded until all leaves are pure. The train accuracy would be 1, but the test accuracy will decrease.

9. How many decision trees do we have to construct if we have to search the two-parameter space, max_depth [1-10] and max_features [4-18]? If we consider 10-fold crossvalidation with the above scenario, how many decision trees do we construct in total?
   If we have to search the two-parameter space, max_depth [1-10] and max_features [4-18], we will have to construct 10*15=**150** trees.
   If we consider 10-fold crossvalidation with the above scenario, we will have to construct **1500** trees in total.

10. For the customer churn prediction task, what is the best choice of k [1-10] for the knearest neighbor algorithm in the 10-fold cross-validation scenario?
    The best choice of k is **9** for the knearest neighbor algorithm in the 10-fold cross-validation scenario (Fitting 10 folds for each of 10 candidates, totalling 100 fits).

11. For MNIST dataset, what was the accuracy of the decision tree [max depth = 5] and Knearest neighbor [K = 10]? What are the best parameters and accuracy for the holdout dataset for decision trees when we used GridSearchCV with 5-fold cross-validation?
    - accuracy of the decision tree [max depth = 5] is 0.6666666666666666
    - accuracy of the Knearest neighbor [K = 10] is 0.975925925925926
    - The best parameters and accuracy for the holdout dataset for decision trees when we used GridSearchCV with 5-fold cross-validation?
      - The best parameters: {'max_depth': 10, 'max_features': 50}
      - Accuracy: 0.8425925925925926