**Final Project Report**

**Problem 1: Car Price Prediction with Linear Regression**

**Description of the Algorithm**

A basic statistical method for modeling and analyzing the relationships between variables is called linear regression. By fitting a linear equation to observed data, it describes the connection between a dependent variable and one or more independent variables. A basic linear regression model's equation is as follows:

**Y = a + bX**

where:

* Y is the dependent variable (target).
* a​ is the y-intercept of the regression line.
* b ​ is the slope of the regression line.
* XX is the independent variable (feature).

When applying multiple linear regression, the formula becomes:

Y = b0​+b1​X1​+b2​X2​+⋯+bn​Xn​

where:

* X1,X2,…,XnX1​,X2​,…,Xn​ are the independent variables (features).
* b1,b2,…,bn​ are the coefficients.

Here is an example code to implement polynomial regression:

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

**Enhancement of Model Performance**

Polynomial features were added to the model to improve it and capture non-linear interactions. Using GridSearchCV, the ideal degree for polynomial features was found.

The polynomial degree that produced the highest model performance was chosen by the grid search after investigating a variety of options.

Scatter Plot of Actual vs. Predicted Prices:

* Without the optimization:

A computer code with text

Description automatically generated with medium confidenceA graph with red and blue dots

Description automatically generated

* Without the optimization:

A computer code with text

Description automatically generated with medium confidenceA graph of red and blue dots

Description automatically generated

**Problem 2: Car Price Classification with Random Forest Classifier**

**Description of the Algorithm**

The Random Forest Classifier is an ensemble learning technique that builds several decision trees during training and produces the individual trees' mode of categorization. It enhances the final outcome by combining the output of several trees. Random Forest has robustness against overfitting, particularly in scenarios with extensive datasets and numerous characteristics.

Important variables consist of:

* n\_estimators: The number of trees in the forest
* max\_depth: The trees' maximum depth.
* min\_samples\_split: The minimum of samples needed to separate an internal node
* min\_samples\_leaf: The minimum of samples necessary for a leaf node to exist.
* bootstrap: Whether bootstrap samples are used when building trees.

The algorithm functions as follows:

1. generating several decision trees using data bootstrap samples.
2. choose a portion of the features at random for every tree.
3. aggregating the predictions of all the trees to determine the final prediction.

Example code to implement Random Forest Classifier:

A screenshot of a computer code

Description automatically generated

In order to find the best-performing model, a systematic testing of different combinations of the hyper-parameters was conducted using GridSearchCV.

The optimization of parameters like max\_depth, n\_estimators, and others resulted in a notable gain in accuracy for the optimized model

Code example for assessing the optimized model:

A screenshot of a computer program

Description automatically generated

**Problem 3: Diamond Clustering with KMeans and PCA**

**Description of the Algorithm**

KMeans is a widely used clustering algorithm that partitions data into k clusters, where each data point belongs to the cluster with the nearest mean. Until convergence, the algorithm repeatedly updates the cluster centers and assigns points to clusters.

The KMeans algorithm's steps are as follows:

1) Randomly initialize the centroids of k clusters.

2) Assign the closest cluster centroid to every data point.

3) Determine the mean of every point in each cluster to update the centroids.

4) Continue with steps 2 and 3 until there is no discernible change in the centroids.

PCA (Principal Component Analysis): This dimensionality reduction method reduces the number of dimensions in the data while preserving the most significant variation by transforming the data into a new coordinate system. This is very helpful when displaying data that has many dimensions.

In this research, the diamond data was clustered using KMeans, and the dimensionality was reduced for visualization using PCA.

Example code to implement KMeans and PCA:

A screenshot of a computer code

Description automatically generated

Using the elbow method we managed to choose the correct amount of cluster: A graph of a number of clusters

Description automatically generated

Here is the result when we choose 2 clusters:

A diagram of a cluster of diamonds

Description automatically generated

In order to find the best-performing model, a systematic testing of different combinations of the hyper-parameters was conducted using GridSearchCV.

The optimization of parameters like max\_depth, n\_estimators, and others resulted in a notable gain in accuracy for the optimized model.