Bitcoin is a digital virtual currency. In recent years, the Bitcoin market has developed rapidly and has been recognized as a new type of gold by many investors. It may replace gold as a hedge against inflation and become a new type of investment asset for financial management. The investment relationship with gold has more and more important research value and practical significance. So we were asked by a trader to develop a model that uses daily price flow data from September 11, 2016 - September 10, 2021 to determine whether a trader needs to buy, hold or sell investments on a daily basis Assets in a portfolio consisting of USD, Gold and Bitcoin. Therefore, the purpose of our report is to establish a predictive model of investment trading strategies to help traders maximize investment profits. At the same time, we will also provide traders with some trading strategies and suggestions to help him better understand the financial market and reduce the financial risk of his investment. To this end, we established two models: Model 1: a linear regression prediction model based on machine learning; Model 2: KNN algorithm and QDA analysis combined model.

For Model I, first we got the pricing data of gold and bitcoin within 5 years, and then we preprocessed the data, and used machine learning to train and validate the data to predict the price fluctuations and their price fluctuations in the next n days. The average value, according to the results displayed, determine the linear regression method to fit the data. Finally, on this basis, in order to obtain the best investment strategy, we use the future price to change the ratio of gold and Bitcoin, and strive to get the highest return. Proportion, that is, using the formula derived from the price change weight according to the Logistics function to determine the final optimal daily trading strategy, thereby calculating the investment value of the initial amount.

For Model II, We used a hybrid combined model of the KNN algorithm and QDA analysis to try to assess its goodness of fit. Since the most critical data for determining the final ratio is the estimated future price fluctuations of gold and bitcoin, we use feature similarity to judge the degree of fit between the estimated data of gold and bitcoin and the real data, and then use the KNN algorithm to determine the degree of fit between the estimated data and the real data. QDA analysis makes multiple judgments to verify whether our forecast is accurate enough to provide the greatest possible return. Finally, we calculate the goodness of fit between gold and Bitcoin to help verify our estimated price. Whether the fluctuations are accurate enough.

Eventually, we also conducted a sensitivity analysis on the investment strategy we established, by changing the size of the initial cost of the linear change, and expanding the range of change from $1,000 to $10,000, to observe the change in the final benefit, and then obtain the change through Python. A straight line is drawn, and this scheme is what we use to try to verify that costs and benefits are highly linearly related. The results also fully show that our initial amount has a positive correlation with the final return, and it also reflects that our model is relatively stable and reliable, which can increase investors' great investment confidence, and also has room for improvement.

Keywords: Virtual currency; Gold; Trading straregy; Machine learning;Best yield;KNN algorithm;QDA analysis;Goodness of fit

1 Introduction

1.1 Problem Background

市场交易员经常买卖波动性资产，是一种具有风险性的投资行为，其波动性通常以一段时间涨落的标准差来衡量。在这些可买卖的波动性资产中，黄金和比特币扮演了重要的角色。

随着社会经济的快速发展，黄金投资已经成为投资者投资组合中的一个重要组成部分。因此，黄金市场在金融市场上为投资者开辟了丰富的投资渠道，提高了金融市场的投资功能。对于比特币而言，是一种总量恒定 2100 万的数字货币，和互联网一样具有去中心化、全球化、匿名性等特性。比特币的流通性和有限性，决定了比特币是可以充当相当于一般等价物的货币功能，只不过这种衡量不是像黄金一样的实体衡量，而是一种数字产品。

(a) Gold

(b) Bitcoin

然而黄金和比特币的性质和投资吸引力却不大相同，因此如何对两者做出正确的投资战略，成为了大众所关注的问题。基于此，交易员向我们团队请求了帮助，交易员要求我们开发一种模型，仅使用 2016 年 9 月 10 日-2021 年 9 月 10 日的每日价格流来确定交易员每天是否应该购买、持有或出售其投资组合中的资产，其组合包括现金、黄金和比特币。

1.2 Restatement of the Problem

在上述问题中，我们得知交易的初始状态为 $1000，假设 α gold = 1%， α bitcoin = 2%。持有资产没有成本。考虑到问题陈述中确定的背景信息和限制条件，我们需要解决以下问题：

- 开发一个模型，根据价格数据提供每日最佳交易策略，并且得出截至 2021 年 9 月 10 日，最初的 1000 美元将得到多少投资回报。
- 提供相关证据能够清楚准确地证明建立的模型提供了最佳投资策略。
- 分析建立的模型与交易成本的关联程度，以及交易员假定的交易成本如何影响策略和结果。
- 以备忘录的形式向交易员传达我们所建立的模型、投资策略，全面展示我们得到的结果。

1.3 Our work

2 Assumptions and Justifications

为了简化问题，我们做了以下基本假设，每一个假设都是适当合理的。

- **Assumption 1: 假设交易者仅在黄金交易日进行交易。**

Justification: 因为黄金仅在开市日上进行交易，这说明周末或者节假日时期黄金的交易状态会保持持有状态，因此我们假设交易发生时期均在黄金开市日进行。

- **Assumption 2: 假设交易者每天只进行一次交易。**

Justification: 交易员的要求中没有限制每天的交易次数，为了使模型能够达到更高的预测率和精确度，我们假设交易者每天只能进行一次交易，最大程度地减小相关误差。

- **Assumption 3: 假设每一次交易均采用最小交易成本方式来进行交易。**

Justification: 在交易员所提出的要求中我们得知，每笔交易的佣金成本为交易金额的 α %，并且 α gold = 1%， α bitcoin = 2%，因此我们需假设每一次交易时的交易成本为最小状态，这样才能使最终 1000 美元的价值最大。

- **Assumption 4: 假设美金、黄金以及比特币的交易过程中交易数额存在小数点。**

Justification: 黄金和比特币作为流通的货币均有易于分割的特性，比特币可以分割到小数点后八位，也就是 1 聪，而黄金虽然也能分割，但相较于比特币而言还是麻烦的多，一次我们假设两者在交易过程中交易数额能精准到相同的小数点。

3 Notations

Table 1: Notations used in this paper

Symbol	Description	Unit
PCT_change	Past price fluctuation	%
HL_PCT	Maximum price difference in the past	%
$present\_crash$	Cash held after the transaction	$
$present\_gold$	Post-trade gold holdings	oz.t
$present\_bitcoin$	Hold bitcoin after the transaction	BTC
δ_1	Change in the estimated price of gold (15 days later)	$/oz.t
δ_2	Change in the estimated price of Bit (after 15 days)	$/BTC
$thePriceOfGold$	The current price of gold	$
$thePriceOfBitcoin$	The current price of bitcoin	$
α_1	Gold transaction commission	%
α_2	Bitcoin transaction commission	%

4 Model Preparation

在这一部分中，我们将简要概述我们用于建立后续描述的模型的数据。

4.1 Given Data

我们的模型是围绕如何对黄金和比特币进行投资以获得最大价值来建立的，在此基础上我们获得了 2016 年 9 月 10 日至 2021 年 9 月 10 日中黄金和比特币的每日定价数据，这些数据分别来自伦敦金银市场协会和 NASDAQ，从 LBMA-GOLD.csv 和 BCHAIN-MKPRU.csv 这两个定价数据文件中，我们不但了解到了黄金和比特币每日的价格波动范围以及涨幅趋势，还可以知道比特币可以每天交易，而黄金仅在开市日交易，这对于我们来说是一个关键信息。

由于提供的数据量较大且不直观，我们可以在后续建模过程中，将数据更加可视化。

5 Model Ⅰ：基于机器学习的线性回归预测模型

5.1 方法概述

机器学习是一类算法的总称，其目的是为了从大量的历史数据中挖掘出其中中隐含的规律，以便用于预测或者分类。具体而言，机器学习可以看作是寻找一个函数，输入是样本数据，而输出的则是期望的结果，只是函数过于复杂，以至于不能形式化表达。机器学习模型可以分为回归模型、分类模型和结构学习模型。为了完成交易员所提出的要求，我们将根据交易员提供的黄金和比特币的数据样本建立回归模型来进行预测分析，以此来得出我们的投资策略。

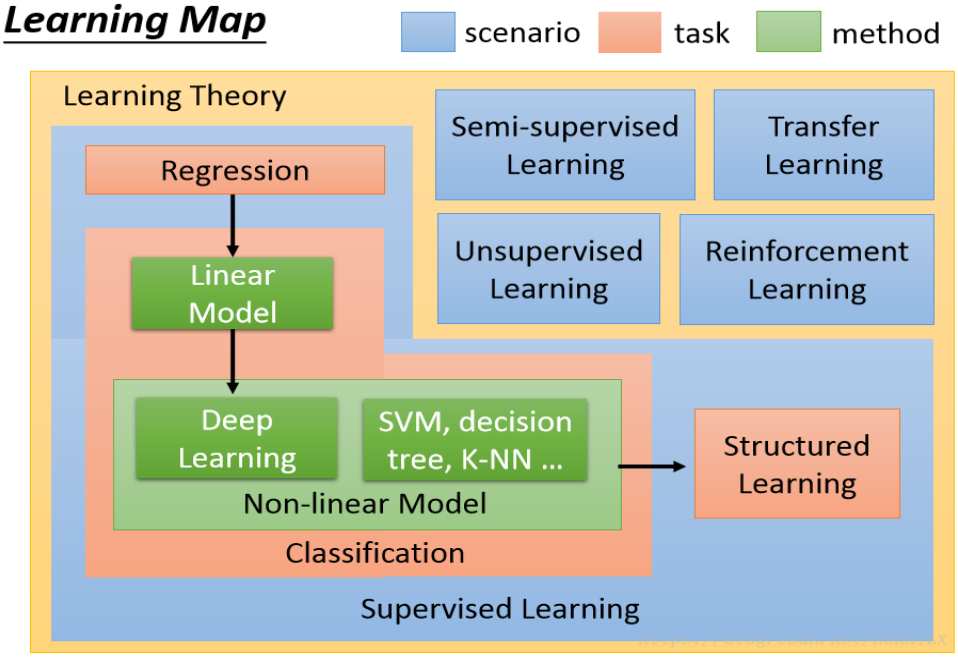


Figure3：机器学习流程图

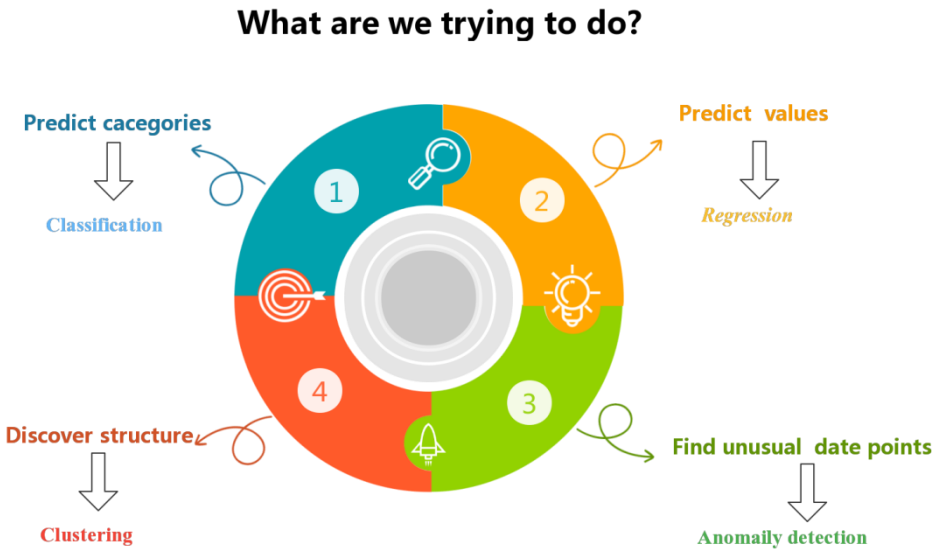


Figure4:机器学习分类图

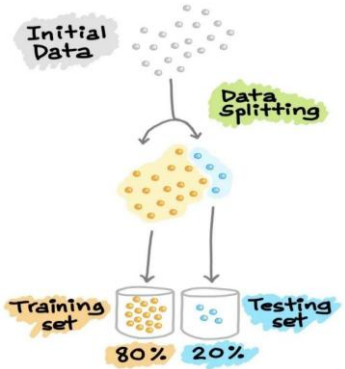
5.2 数据预处理

基于建立机器学习回归预测模型基础上，我们在进行数据分析之前，必须要保证数据的准确性和可用性。如果基于不可靠的数据进行机器学习，则无法得出准确的预测结果。从交易员提供的数据文件可知，黄金只在开市日进行交易，因此我们对 LBMA-GOLD.csv 文件中的空白行进行了删除处理。除此之外，为了保证数据的一致性，减小预测误差，我们同时删除了黄金停市日时比特币对应的定价数据，使得模型的预测能够更加精准。

5.3 数据分割

在我们建立的机器学习线性回归预测模型的开发过程中，希望训练好的模型能在新的、未见过的数据上表现良好。为了模拟新的、未见过的数据，我们对黄金和比特币的定价数据进行数据分割，从而将其分割成两部分。其中，第一部分是较大的数据子集，用作训练集，占原始数据的 90%，而第二部分为较小的子集，即剩下的 10%用作测试集。

接下来，我们利用训练集建立预测模型，然后将这种训练好的模型应用于测试集上进行预测，以便根据模型在测试集上的表现来选择最佳模型。



5.4 线性回归模型

5.4.1 回归问题

回归是用于预测输入变量和输出变量之间的关系，特别是我们给定的输入变量的值发生变化时，输出变量的值随之发生变化的变化。回归模型正是表示输入变量到输出变量之间映射的函数。此类问题的学习等价于函数拟合，即选择一条函数曲线来使得其很好地拟合我们给出的 2016 年 9 月 10 日-2021 年 9 月 10 日这段时间内黄金和比特币的定价数据，来预测未来的数据价格。

5.4.2 线性回归模型描述

在我们建立的线性回归分析中，包括了两个自变量，且因变量和自变量之间是线性关系。我们选择了黄金与比特币当天之前的价格波动以及最大与最小的差值作为影响因子，其中用到的多元线性回归中的公式如下：

$$h_\theta(X^i) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \tag{1}$$

在这里，我们默认为x_0总是等于 1。为了方便表示，我们把假设函数可以写成向量的形式：

$$h_\theta(X^i) = \theta^{\mathrm{T}} X \tag{2}$$

其中，

$\theta = [\theta_0, \theta_1, \ldots, \theta_n]^{\mathrm{T}}$	(1)
$X = [1, x_1, x_2, \ldots, x_n]^{\mathrm{T}}$	(1)

影响因子求解公式如下：

$$\begin{aligned} Y &= \theta_1 * X_1 + \theta_2 * X_2 + \complement Gold(ETF)price \\ &= \theta_1 * PCT\_change + \theta_2 * HL\_PCT + c \end{aligned} \tag{3}$$

为了选出最合适的线性回归模型，我们还是用到了损失函数，我么的目的是找出使损失函数最小的向量θ。我们在给定的黄金和比特币的数据文件中找出一条线去拟合它，那么我先假设这个线的方程，然后把数据点代入假设的方程得到观测值，求使得实际值与观测值相减的平方和最小的参数，公式如下：

$$J(\theta) = \frac{1}{2m}\sum_{i=0}^{m}(h_\theta(X^i) - y^i)^2 = \frac{1}{2m}(X_\theta - y)^{\mathrm{T}}(X_\theta - y)$$

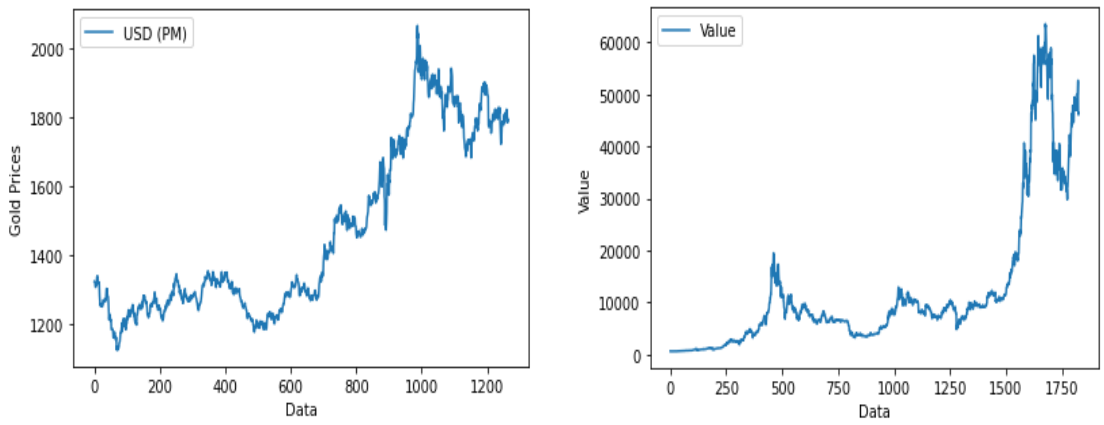
除此之外，为了获得局部最优解，机器学习的过程中还采用到了梯度下降算法，整个过程使用的公式如下：

$$\frac{\partial}{\partial\theta}J(\theta) = \frac{\partial}{\partial\theta}\frac{1}{2}\sum_{i=1}^{m}(h_\theta(x) - y)^2 = (h_\theta(x) - y)x^{(i)}$$

5.5 结果

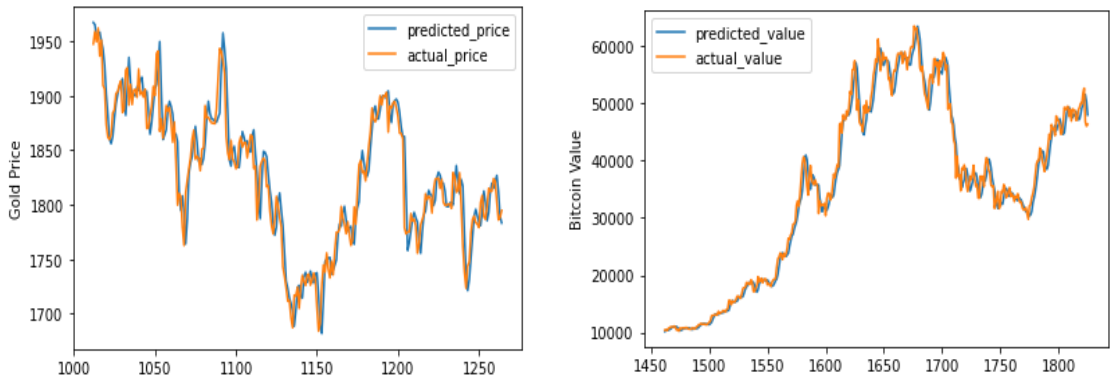
5.5.1 预测结果

在 python 的帮助下，可以实现上述建模过程，通过我们设置的变量可以预测出黄金与比特币的定价数据以及波动范围。经过我们的拟合可以看出，其预测值的准确率较高，结果如下图所示：



(a) 黄金预测数据　　(b) 比特币预测数据

Figure 10: 黄金和比特币的预测数据



(a) 黄金数据拟合　　(b) 比特币数据拟合

Figure 10: 黄金和比特币的拟合数据

5.5.2 拟合优度检验

拟合的过程其实是寻找能很好地温和黄金和比特币数据序列的数学模型的过程而在拟合好数据之后，为了更好地对我们的预测准确率进行评价，我们采用了拟合优度检验方案，借助检验概率的概念来评价黄金和比特币数据拟合的质量。我们利用卡方统计量进行统计显著性检验，其公式如下：

$$X_{df}^2 = \sum_{i=1}^{k} \frac{(O_i - T_i)^2}{T_i} \tag{4}$$

计算得出黄金的拟合优度为 89.44%，比特币的拟合优度为 98.43%，拟合精度较为准确。

5.5.3 价格变化权重确定最佳策略

我们通过预测黄金和比特币后十五天的价格变化，取其平均值作为预计金额，我们根据二者预估价格平均值，以二者预估价格变化量为权，通过使用 Logistics 函数，再根据下面的公式:

$$present\_crash : present\_gold : present\_bitcoin = logistic(\delta_1 + \delta_2) : logistic(\delta_1) : logistic(\delta_2)$$

其中δ是预估平均价格减去当前的价格，δ_1是表示为黄金的价格，δ_2表示为比特币的价格。上式中，我们通过控制每日黄金和比特币的比例来确定投资策略。

$$\begin{aligned} present\_crash + present\_gold * thePriceOfGold + present\_bitcoin \\ * thePriceOfBitcoin \\ = previous\_crash + previous\_gold * thePriceOfGold \\ + previous\_bitcoin * thePriceOfBitcoin - \alpha_1 \\ * \frac{previous\_gold - present\_gold}{thePriceOfGold} - \alpha_2 \\ * \frac{previous\_bitcoin - present\_bitcoin}{thePriceOfBitcoin} \end{aligned}$$

（注：根据题中所给数据$\alpha_1 = 0.01, \alpha_2 = 0.02$）

根据下述公式，我们再辅以判断:

$$\begin{aligned} crash\_all < present\_gold + avarage\_gold * thePriceOfGold + avarage\_bit \\ * thePriceOfBitcoin \end{aligned}$$

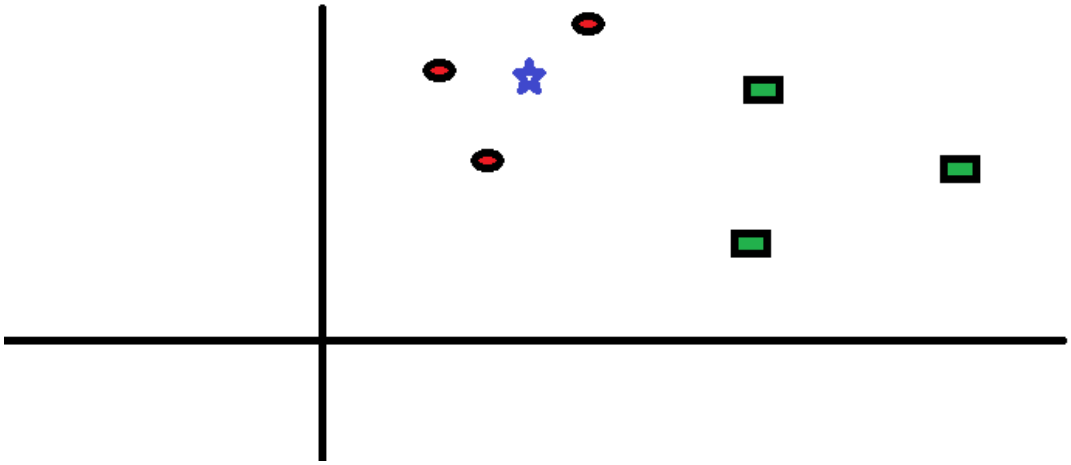
如果上式成立，交易员就可以在黄金开市日进行交易，否则将不进行交易。通过该算法，我们成功得出了最终结果。

6 Model II: KNN 算法及 QDA 分析组合模型

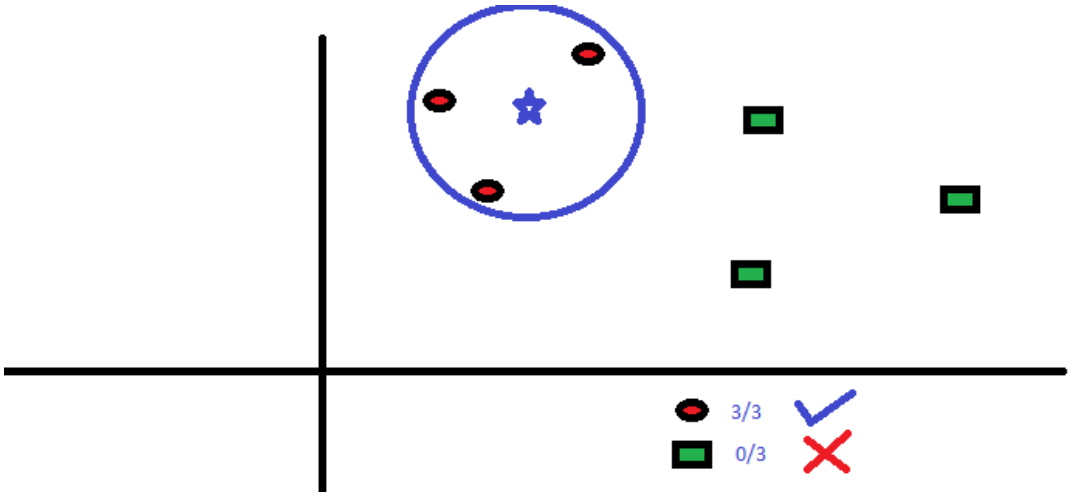
为了保证黄金和比特币预测出来的新数据与原来数据集中的点相同，我们决定采用 KNN 算法利用特征相似性来预测二者的数据算法利用特征相似性来评估二者预测数据准确程度。除此之外，我们还用到了 QDA 分析，通过我们的组合模型来判断我们所建议的投资组合方式是否具有最大化效益，并为交易者提供相关证据。

6.1 KNN 算法运用

首先我们可以用两张简单的图片来理解 KNN 算法如何帮助我们实现目标，以下是红色圆圈（RC）和绿色方块（GS）的分布。

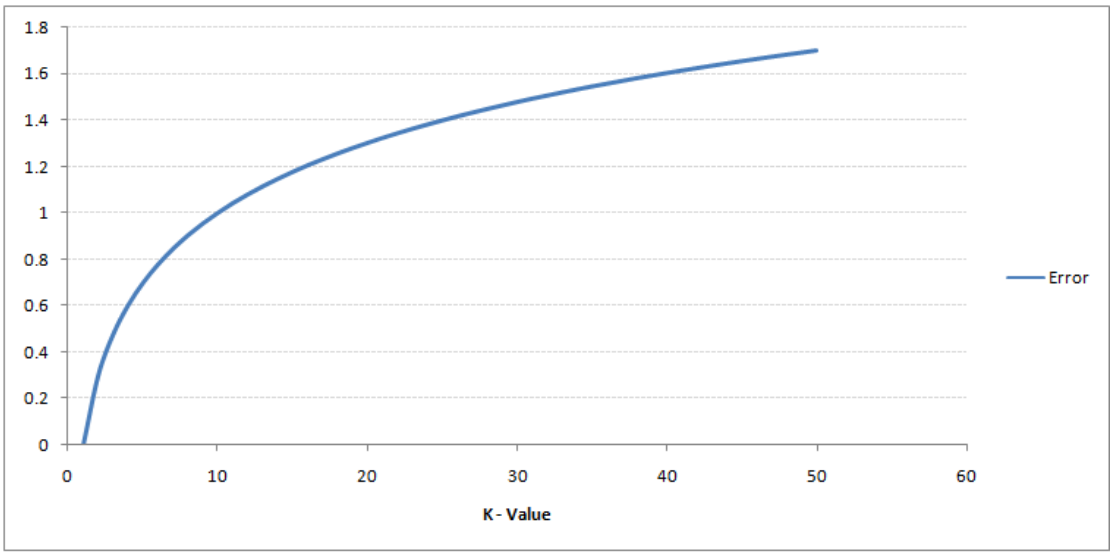


我们打算找出蓝星（BS）的等级，BS 可以是 RC 或 GS。"K"是 KNN 算法中我们所希望从中选出的最近邻居。

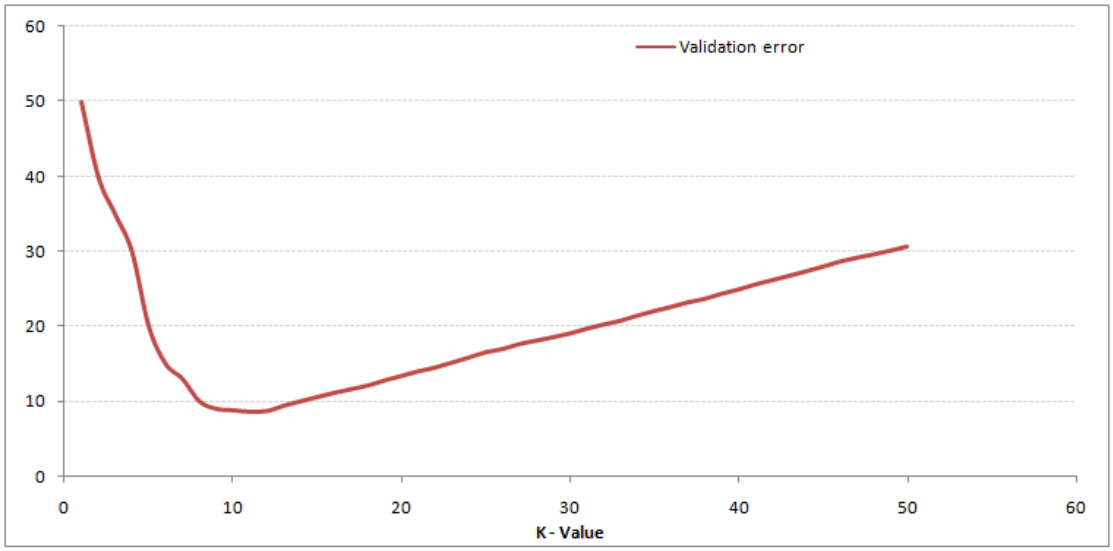


6.1.1 选择合适的 K 值

在这个过程中，我们要选择合适的 K 值来对黄金和比特币进行预测，首先我们要训练错误率和验证错误率，这两者都需要我们访问不同 K 值的两个参数，下图是具有可变 K 值的训练错误率曲线，我们可以看到边界随着 K 值的增加而变得越来越平滑。



由图可知，对于训练样本，K=1 处的错误率始终为零。这是因为最接近任何训练数据点的点是自身。因此，当 K=1 时，预测始终是准确的。如果验证误差曲线相似，我们选择的 K 将是 1。以下是具有不同 K 值的验证误差曲线：



6.1.2 得出结果

上述过程使得我们的目标更加清晰。在 K=1 时，我们过度拟合了边界。因此，错误率最初降低并达到最小值。在最小点之后，它随着 K 的增加而增加。若要获取 K 的最优值，可以将训练和验证与初始数据集隔离开来。现在绘制验证误差曲线以获得 K 的最优值。此 K 值就可以应用于黄金和比特币的预测上。

6.2 QDA 分析

我们在组合模型中所用到的 QDA 可以认为是非参数学习法 KNN 和具有线性分类边界的 LR 及 LDA 方法的折中，QDA 假设分类边界是二次的，所以它能够比线性模型更准确的为我们要解决的问题建立模型。同时由于其二次边界的额外假设，当样本量较小时，能够优于 KNN 方法。

6.2.1 贝叶斯模型的建立

根据贝叶斯模型，其公式如下：

$$P(C_1|x) = \frac{p(C_1,x)}{p(x)} = \frac{p(C_1)p(x|C_1)}{p(C_1)p(x|C_1) + p(C_2)p(x|C_2)}$$

我们只要计算对$p(C_1)$、$p(x|C_1)$、$p(C_2)$和$p(x|C_2)$进行建模并计算求出$P(C_1|x)$即可，而$P(C_1)$属于先验概率，直观的看应该等于$\frac{N_{c1}}{N}$。

6.2.2 假定服从正态分布

对于$p(x|C_1)$，我们假定服从正态分布，假设 x 是 D 维向量，则正态分布的形式如下：

$$\frac{1}{2\pi^{D/2}|\Sigma|^{1/2}} e^{\left\{-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right\}}$$

其中参数μ_k和 Σ 是可以通过训练集的最大释然来求解。

6.2.3 最大释然求解

从上述步骤中我们可以得出：$p(x|C_1)$是一个指数函数，$P(C_1)$是一个常数，所以两者的乘集也是一个指数函数，那么：

$$\ln(p(C_1|x)) = (x - \mu_1)^T\Sigma^{-1}(x - \mu_1) + C$$

，同理$p(x|C_2)$也是类似的形式，故:

$$\ln(\frac{p(C_1|x)}{p(C_2|x)}) = (x - \mu_1)^T\Sigma^{-1}(x - \mu_1) - (x - \mu_2)^T\Sigma^{-1}(x - \mu_2) + C$$

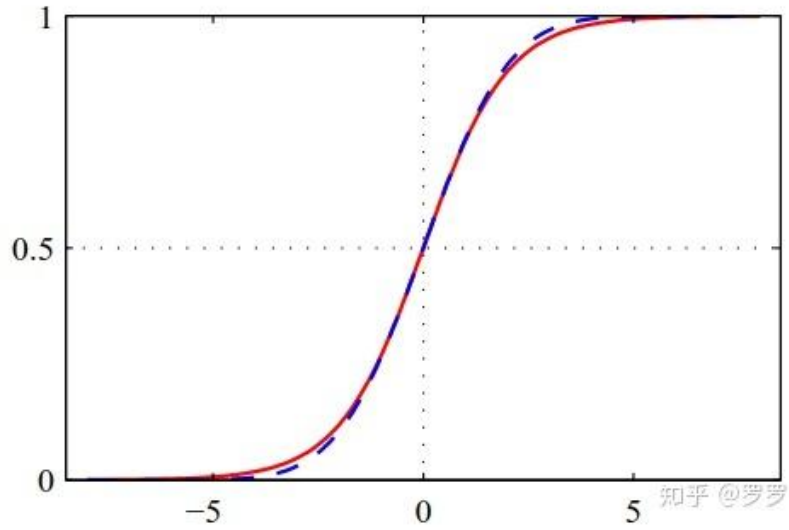
由于二个类别的协方差矩阵Σ相同，使得 x 的二次项被消去，所以只包含一次项的形式，故$\ln(\frac{p(C_1|x)}{p(C_2|x)})$就是一个 x 的线性函数，可以写成$-(\omega^T x + \omega_0)$的形式。

由于：

$$P(C_1|x) = \frac{p(C_1,x)}{p(x)} = \frac{p(C_1)p(x|C_1)}{p(C_1)p(x|C_1) + p(C_2)p(x|C_2)} = \frac{1}{1 + e^{\ln(\frac{p(C_1|x)}{p(C_2|x)})}}$$

$$= \frac{1}{1 + e^{-(\omega^T x+\omega_0)}}(\omega^T x + \omega_0)(\omega^T x + \omega_0)$$

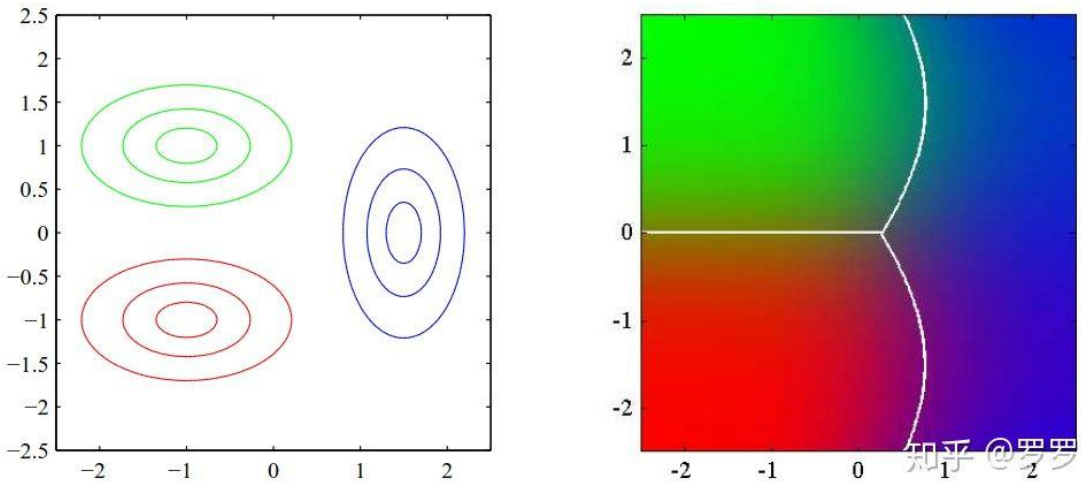
由此发现$P(C_1|x)$可以表示成 sigmod 函数的形式。sigmoid 的函数图像为:



横坐标为$(\omega^T x + \omega_0)$，纵坐标为$P(C_1|x)$。

以上，可以直观的认为，可以通过训练集，求取参数 w 和ω_0，使得对于类别一的绝大部分数据 x 都有$\omega^T x + \omega_0 > 0$，对于类别二的绝大部分数据都有$\omega^T x + \omega_0 < 0$，这和 fisher 的判别方法（即类之间的距离远，类间方差小的）都是相同的模型,只是求解的过程不同。到底哪个方法好了？还要测试测试。

所有以上的论述中，都是**假定了各个类别的协方差矩阵Σ相同**，所以两个类别的判别边界是条直线（处于两个类判别边界的点是$P(C_1|x) = P(C_2|x)$。如果各类别的协方差矩阵都不相同，则判别边界不是直线，此情况下称为**二次判别**。



三个类别的类条件概率密度，每个都是高斯分布，分别用红色、绿色、蓝色表示，其中红色和绿色的类别有相同的协方差矩阵。右图给出了对应的后验概率分布，决策边界也被画出。注意，具有相同协方差矩阵的红色类别和绿色类别的决策边界是线性的，而其他类别之间的类别的决策边界是二次的。

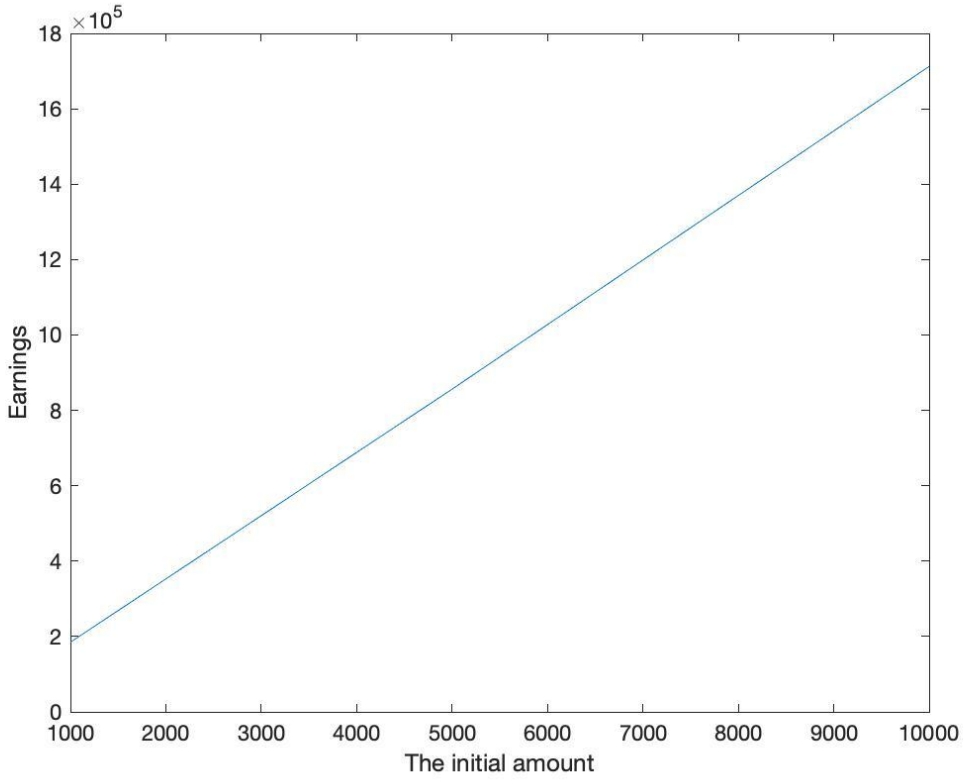
Score 函数确定投资组合

6.3 Score 函数确定投资组合

在最后，我们采用 Score 函数来确定我们的投资组合方案，其中 Score 函数是我们在运用 Python 求解使用到的内置函数。在我们得出的结果中，我们还使用决定系数来反映黄金和比特币中，我们设置的自变量对因变量的的解释程度。决定系数反应了 y 的波动有多少百分比能被 x 的波动所描述，即表征依变数 Y 的变异中有多少百分比,可由控制的自变数 X 来解释。若解释程度越高，自变量引起的变动占总变动的百分比也就越高，我们的投资组合也就越成功。

7 一 Test the Model

我们利用我们的模型，通过更改初始金额的大小，得到了一系列数据，其变化范围从 1000 美金到 10000 美金，通过 Python 我们最终绘画出了一条笔直的直线，图像如下：



这充分说明我们的初始金额与最终收益具有完全正相关关系。

7.1 Sensitivity Analysis

In section 6.3.2, two exogenous factors are introduced to estimate the parameters of the profit evaluation model of fishing companies: social profit rate and average

navigation dis- tance. Therefore, the relationship between the final survival rate of fishing companies and these two factors is approximated by first-order difference:

()

Therefore, the calculation results are shown in Figure 15.

It is indicated that the ultimate survival rate of fishing companies increased with the social profit margin, which reflects the feedback effect of social development on fishing companies; correspondingly, the ultimate survival rate of fishing companies is also increase with the average navigation distance, which shows the importance of long-distance navigation ability in promoting the profitability of fishing companies. The trend of the model obtained by sensitivity test is consistent with the actual situation, which also proves the rationality and robustness of the profit evaluation model of fishing companies.

7.2 一.2 Robustness Analysis

For model 2, it is significant to consider whether the final migration states can be stable under different initial distribution samples. Therefore, the migration of fish based on model 2 under different initial distribution conditions is simulated, and the final distribution for -th is expressed as . Then the degree of fit between any two final distributions is calculated by:

()

And is combined into pair series to draw a Q-Q plot as

It can be seen that the scattered points of the figure are distributed on the straight line, which shows that the overall fit situation is stable. It means that the similar final fish distribution can be obtained on the premise of different initial fish distribution, which verifies the decisive role of temperature change on fish migration and reflects the stability of the model.

Appendices

Appendix 1
Introduce: Tools and software
Paper written and generated via Office 2019. Graph generated and calculation using Python 3.7.0.

Appendix 2
Introduce: 投资策略模型代码

```
import sklearn
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
```

```
from matplotlib import style
import matplotlib as mpl
import datetime

gold_df = pd.read_csv('Python/LBMA-GOLD.csv')
bitcoin_df = pd.read_csv('Python/BCHAIN-MKPRU.csv')
combination_df = pd.merge(bitcoin_df, gold_df, on=['Date'], how='outer')

# Suppose on day 2 the trader allocates assets as follows
crash = 500.0
gold = 250.0 / 1324.6
bitcoin = 250.0 / 609.67
holdings = [crash, gold, bitcoin]
crash_all = 1000

# i is days of transaction.
for i in range(10, 1810):
    print('The ', i, 'th has begun !')
    print()
    # combination_df_temp = combination_df.fillna(value=0, inplace=True)
    if combination_df.iloc[i].at['USD (PM)'] == 0:
        continue
    gold_df = pd.concat([pd.DataFrame(combination_df[:i]['Date']),
                         pd.DataFrame(combination_df[:i]['USD (PM)'])], axis=1, join='outer')
    bitcoin_df = pd.concat([pd.DataFrame(combination_df[:i]['Date']),
                            pd.DataFrame(combination_df[:i]['Value'])], axis=1, join='outer')
    gold_df = gold_df.dropna()
    bitcoin_df = bitcoin_df.dropna()
    # Get moving average
    gold_mavg = gold_df['USD (PM)'].shift(1).rolling(window=i+1).mean()
    bitcoin_mavg = bitcoin_df['Value'].shift(1).rolling(window=i+1).mean()
    gold_df = gold_df.dropna()
    bitcoin_df = bitcoin_df.dropna()

    # earnings
    earnings_gold = gold_df['USD (PM)'] / gold_df['USD (PM)'].shift(1) - 1
    earnings_bit = bitcoin_df['Value'] / bitcoin_df['Value'].shift(1) - 1

    dfreg_gold = gold_df.loc[:, ['USD (PM)']]
    dfreg_gold['PCT_change'] = gold_df['USD (PM)'].pct_change() * 10000
    for j in list(gold_df.index):
        dfreg_gold.loc[j, 'HL_PCT_gold'] = float(dfreg_gold[['USD (PM)']].max(
        )) - float(dfreg_gold[['USD (PM)']].min()) / dfreg_gold.loc[j, 'USD (PM)'] * 1000
    dfreg_bit = bitcoin_df.loc[:, ['Value']]
```

```
dfreg_bit['PCT_change'] = bitcoin_df['Value'].pct_change() * 100000
for j in list(bitcoin_df.index):
    dfreg_bit.loc[j, 'HL_PCT_bit'] = float(dfreg_bit[['Value']].max(
    )) - float(dfreg_bit[['Value']].min()) / dfreg_bit.loc[j, 'Value'] * 100000
dfreg_gold['Date'] = combination_df[['Date']]
dfreg_bit['Date'] = combination_df[['Date']]
# dfreg_gold.plot()
# Drop missing value
dfreg_gold.fillna(value=-99999, inplace=True)
dfreg_bit.fillna(value=-99999, inplace=True)
# The separation of data（training：testing = 9:1）
# We want to separate 10 percent of the data to forecast
forecast_out_gold = int(math.ceil(0.1 * len(dfreg_gold)))
forecast_out_bit = int(math.ceil(0.1 * len(dfreg_bit)))
forecast_col_gold = 'USD (PM)'
forecast_col_bit = 'Value'
dfreg_gold['label'] = dfreg_gold[forecast_col_gold].shift(
    -forecast_out_gold)
dfreg_bit['label'] = dfreg_bit[forecast_col_bit].shift(-forecast_out_bit)
X_gold = np.array(dfreg_gold.drop(['label', 'Date'], 1))
X_bit = np.array(dfreg_bit.drop(['label', 'Date'], 1))
# Scale the X so that everyone can have the same distribution for linear regression
X_gold = sklearn.preprocessing.scale(X_gold)
X_bit = sklearn.preprocessing.scale(X_bit)
# Finally We want to find Data Series of late X and early X (train) for model generation and evaluation
X_lately_gold = X_gold[-forecast_out_gold:]
X_lately_bit = X_bit[-forecast_out_bit:]
X_gold = X_gold[:-forecast_out_gold]
X_bit = X_bit[:-forecast_out_bit]
# Separate label and identify it as y
y_gold = np.array(dfreg_gold['label'])
y_gold = y_gold[:-forecast_out_gold]

y_bit = np.array(dfreg_bit['label'])
y_bit = y_bit[:-forecast_out_bit]
X_gold_test = X_gold[-forecast_out_gold:]
X_bit_test = X_bit[-forecast_out_bit:]
y_gold_test = y_gold[-forecast_out_gold:]
y_bit_test = y_bit[-forecast_out_bit:]
# Linear regression
clfreg_gold = LinearRegression(n_jobs=-1)
clfreg_gold.fit(X_gold, y_gold)
clfreg_bit = LinearRegression(n_jobs=-1)
clfreg_bit.fit(X_bit, y_bit)
```

```
# Quadratic Regression 2
clfpoly2_gold = make_pipeline(PolynomialFeatures(2), Ridge())
clfpoly2_gold.fit(X_gold, y_gold)
clfpoly2_bit = make_pipeline(PolynomialFeatures(2), Ridge())
clfpoly2_bit.fit(X_bit, y_bit)
# Quadratic Regression 3
clfpoly3_gold = make_pipeline(PolynomialFeatures(3), Ridge())
clfpoly3_gold.fit(X_gold, y_gold)
clfpoly3_bit = make_pipeline(PolynomialFeatures(3), Ridge())
clfpoly3_bit.fit(X_bit, y_bit)
clfknn_gold = KNeighborsRegressor(n_neighbors=2)
clfknn_gold.fit(X_gold, y_gold)
clfknn_bit = KNeighborsRegressor(n_neighbors=2)
clfknn_bit.fit(X_bit, y_bit)
# Forecast the price of the next days
days = 15
last_date_gold = pd.to_datetime(gold_df['Date'])
last_unix_gold = last_date_gold
next_unix_gold = last_unix_gold + datetime.timedelta(days=1)
for j in range(1, days):
    next_date_gold = next_unix_gold
    next_unix_gold += datetime.timedelta(days=1)
    dfreg_gold.loc['Date'] = [
        np.nan for _ in range(len(dfreg_gold.columns)-1)]+[j]
dfreg_gold = dfreg_gold.drop('Date')
bitcoin_df = combination_df.dropna()
bitcoin_df = bitcoin_df.drop('USD (PM)', 1)
bitcoin_df = bitcoin_df.reset_index()
bitcoin_df = bitcoin_df.drop('index', 1)
dfreg_bit = bitcoin_df
last_date_bit = pd.to_datetime(bitcoin_df['Date'])
last_unix_bit = last_date_bit
next_unix_bit = last_unix_bit + datetime.timedelta(days=1)
for j in range(1,days):
    next_date_bit = next_unix_bit
    next_unix_bit += datetime.timedelta(days=1)
    dfreg_bit.loc['Date'] = [
        np.nan for _ in range(len(dfreg_bit.columns)-1)]+[j]
dfreg_bit = dfreg_bit.drop('Date')
combination_df_temp = combination_df.fillna(value=0.0)
if combination_df_temp.iloc[i].at['USD (PM)'] == 0.0:
    continue
# Calculate daily changes
avarage_gold = dfreg_gold[['USD (PM)']][len(dfreg_gold)-days:].mean()
```

```
    avarage_bit = dfreg_bit[['Value']][len(dfreg_bit)-days:].mean()
    avarage_gold = avarage_gold['USD (PM)']
    avarage_bit = avarage_bit['Value']
    # Define the logistic function
    def logistic(z):
        return 1 / (1 + np.exp(-z))
    # Increase the weight of price changes (Logistics)
    # delta = avarage_gold + avarage_bit
    # avarage_gold = logistic(
    #         avarage_gold - combination_df.iloc[i].at['USD (PM)'])
    # avarage_bit = logistic(avarage_bit - combination_df.iloc[i].at['Value'])
    priceOfGold = combination_df.iloc[i].at['USD (PM)']
    priceOfBit = combination_df.iloc[i].at['Value']
    valueOfDallor = crash
    valueOfGold = gold * priceOfGold
    valueOfBit = bitcoin * priceOfBit
    priceOfGold = combination_df.iloc[i].at['USD (PM)']
    priceOfBit = combination_df.iloc[i].at['Value']
    # y2,y2 is quantity
    # present_valueOfDallor = ((priceOfGold * priceOfBit * (valueOfDallor + valueOfGold + valueOfBit)) - (0.01 * valueOfGold *
priceOfBit + 0.02 * valueOfBit * priceOfGold)) / (
    #         ((avarage_bit * priceOfBit + avarage_gold * priceOfGold + 1) * priceOfGold * priceOfBit) - (avarage_bit * priceOfBit
* priceOfGold + avarage_gold * priceOfGold * priceOfBit))
    # y2,y3 is value
    present_valueOfDallor = ((priceOfGold * priceOfBit * (valueOfDallor + valueOfGold + valueOfBit)) - (0.01 * valueOfGold *
priceOfBit + 0.02 * valueOfBit * priceOfGold)) / (
            ((avarage_bit + avarage_gold + 1) * priceOfGold * priceOfBit) - (avarage_bit * priceOfBit + avarage_gold *
priceOfGold))
    # y2,y3 is quantity
    # presnet_valueOfGold = avarage_gold * present_valueOfDallor * priceOfGold
    # present_valueOfBit = avarage_bit * present_valueOfDallor * priceOfBit

    # y2,y3 is value
    presnet_valueOfGold = avarage_gold * present_valueOfDallor
    present_valueOfBit = avarage_bit * present_valueOfDallor
    avarage_gold = dfreg_gold[['USD (PM)']][len(dfreg_gold)-days:].mean()
    avarage_bit = dfreg_bit[['Value']][len(dfreg_bit)-days:].mean()
    avarage_gold = avarage_gold['USD (PM)']
    avarage_bit = avarage_bit['Value']
    # if crash_all > present_valueOfDallor + avarage_gold * gold + avarage_bit * bitcoin:
    #         continue
    # Total daily value held（$）
    crash_all = present_valueOfDallor + presnet_valueOfGold + present_valueOfBit
    crash = present_valueOfDallor
```

```
        gold = presnet_valueOfGold / combination_df.iloc[i].at['USD (PM)']
        bitcoin = present_valueOfBit / combination_df.iloc[i].at['Value']
        print('You have ', crash_all, ' !')
        print()
holdings = [crash, gold, bitcoin]
print(holdings)
```

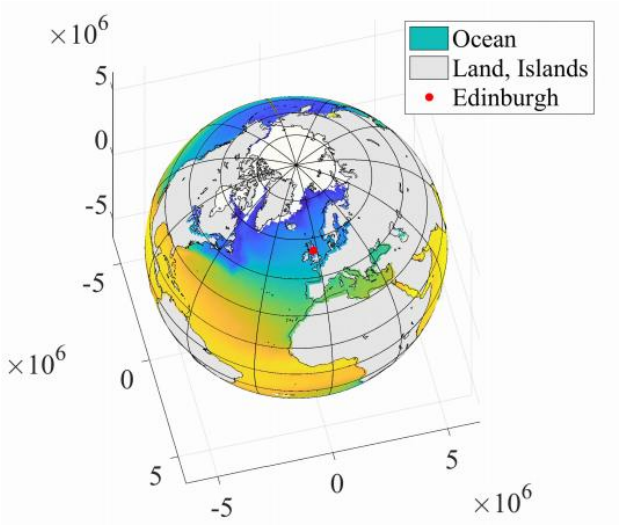


Figure 7: Temperature forecast after 50 years